ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΣΕΡΡΩΝ ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΕΠΙΚΟΙΝΩΝΙΩΝ

ΑΝΑΠΤΥΞΗ ΔΙΚΤΥΑΚΗΣ ΕΦΑΡΜΟΓΗΣ ΚΑΤΑΧΩΡΗΣΗΣ, ΕΠΕΞΕΡΓΑΣΙΑΣ ΚΑΙ ΠΡΟΒΟΛΗΣ ΣΤΟ ΔΙΑΔΙΚΤΥΟ ΤΩΝ ΑΠΟΤΕΛΕΣΜΑΤΩΝ ΤΩΝ ΒΟΥΛΕΥΤΙΚΩΝ ΕΚΛΟΓΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΜΠΟΛΑΡΗ ΑΓΓΕΛΙΚΗ(1451)

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ:

ΒΟΛΟΓΙΑΝΝΙΔΗΣ ΣΤΑΥΡΟΣ

2013

Ευχαριστίες

Με την ολοκλήρωση της παρούσας πτυχιακής εργασίας θα ήθελα να ευχαριστήσω τον κ. Σταύρο Βολογιαννίδη, Επιστημονικό Συνεργάτη του τμήματος Πληροφορικής και Επικοινωνιών του Τ.Ε.Ι. Σερρών και επιβλέποντα της πτυχιακής μου, αρχικά για το ότι δέχτηκε την επίβλεψη της πτυχιακής και αφετέρου για την πολύτιμη καθοδήγηση και βοήθεια που προσέφερε καθ'όλη τη διάρκεια εκπόνησης της πτυχιακής

Επίσης θα ήθελα να ευχαριστήσω τους γονείς μου για την υπομονή και την υποστήριξη που έδειξαν καθ'όλη τη διάρκεια των σπουδών μου αλλά κυρίως κατά τη διάρκεια εκπόνησης της εργασίας αυτής.

Υπεύθυνη Δήλωση

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της πτυχιακής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της, είναι πλήρως αναγνωρισμένη και αναφέρεται στην πτυχιακή εργασία. Επίσης έχω αναφέρει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επίσης βεβαιώνω ότι αυτή η πτυχιακή εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για τις απαιτήσεις του προγράμματος σπουδών του Τμήματος Πληροφορική & Επικοινωνιών του Τ.Ε.Ι Σερρών

Περίληψη

Σκοπός αυτής της πτυχιακής είναι ο σχεδιασμός και η υλοποίηση ενός συστήματος online καταχώρησης ψηφοδελτίων των βουλευτικών εκλογών για τον νομό Σερρών και η έγκυρη και αναλυτική παρουσίαση των αποτελεσμάτων στο ευρύ κοινό με τη χρήση του διαδικτύου.

Παρακάτω θα παρουσιαστούν αναλυτικά οι τεχνολογίες που χρησιμοποιήθηκαν για τον σχεδιασμό και την ανάπτυξη της εφαρμογής. Η εφαρμογή υλοποιήθηκε με τη χρήση εργαλείων ελεύθερου λογισμικού, ανοιχτού κώδικα. Συγκεκριμένα χρησιμοποιήθηκαν: ο διακομιστής ιστοσελίδων Apache, η γλώσσα προγραμματισμού PHP, η οποία είναι κατάλληλη για ανάπτυξη σε περιβάλλον WEB, η βάση δεδομένων MYSQL, η γλώσσα JavaScript μέσω του ExtJS και η χρήση της PHP βιβλιοθήκης GD για τη δημιουργία εικόνων και γραφημάτων. Η εφαρμογή λειτουργεί σε διαδικτυακό περιβάλλον. Απαιτεί όνομα χρήστη και κωδικό πρόσβασης και ανάλογα με τα δικαιώματα του κάθε χρήστη μπορεί να έχει πρόσβαση σε ορισμένα μέρη του προγράμματος.

Περιεχόμενα

Κεφάλαιο 1:Εισαγωγή

1.1 Στόχος

Στόχος της εργασίας είναι η μελέτη, ο σχεδιασμός και η υλοποίηση ενός συστήματος για τη διαχείριση και για την online παρακολούθηση των αποτελεσμάτων των βουλευτικών εκλογών. Η online παρακολούθηση της πορείας των εκλογών είναι μια σημαντική πρόκληση για κάθε νομαρχία ώστε να ενημερώνονται έγκαιρα και έγκυρα οι πολίτες, οι υποψήφιοι, οι δημοσιογράφοι, αλλά και κάθε ενδιαφερόμενος. Παρόμοια εφαρμογή έχει δημιουργήσει η εταιρία **Singular Logic** για το υπουργείο εσωτερικών χωρίς όμως να δείχνει αναλυτικά τις ψήφους που πήρε ο κάθε υποψήφιος.
Αξίζει να αναφερθεί ότι τα επίσημα αποτελέσματα της διαδικασίας των εκλογών ανακοινώνονται από τα πρωτοδικεία των νομών αρκετά αργότερα

Για την υλοποίηση της εφαρμογής θα χρησιμοποιηθούν εργαλεία ελεύθερου λογισμικού ανοιχτού κώδικα

1.2 Δομή πτυχιακής εργασίας

Στο κεφάλαιο 1 περιγράφονται ο σκοπός και η δομή της πτυχιακής εργασίας.

Στο κεφάλαιο 2 περιγράφονται οι Τεχνολογίες και τα εργαλεία ανάπτυξης

Στο κεφάλαιο 3 βρίσκεται η ανάλυση της εφαρμογής και ο κώδικας που χρησιμοποιήθηκε για την υλοποίηση της

Κεφάλαιο 2: Τεχνολογίες και Εργαλεία Ανάπτυξης

2.1 Διαδίκτυο

Το διαδίκτυο αποτελεί πλέον αναπόσπαστο κομμάτι της καθημερινότητας εκατομμυρίων ανθρώπων ανά τον κόσμο.

Με τη χρήση του πρωτοκόλλου επικοινωνίας TCP/IP, μέρος του οποίου είναι και το HTTP, συνδέει ηλεκτρονικούς υπολογιστές ανά τον κόσμο, οι οποίοι βρίσκονται σε ένα κοινό δίκτυο επικοινωνίας, ανταλλάσοντας μεταξύ τους μηνύματα-πακέτα με τη χρήση πρωτοκόλλων σε επίπεδο λογισμικού.

Οι χρήστες αποκτούν πρόσβαση σε έναν ατέλειωτο όγκο πληροφοριών καθιστώντας το το πιο δημοκρατικό μέσο μαζικής επικοινωνίας. Οι πιο διαδεδομένες υπηρεσίες του διαδικτύου είναι ο παγκόσμιος ιστός, το ηλεκτρονικό ταχυδρομείο και η διαμοίραση αρχείων.

2.2 Προγράμματα περιήγησης

Με τον όρο πρόγραμμα περιήγησης (web browser) αναφερόμαστε στις εφαρμογές που χρησιμοποιούμε για να δούμε σελίδες όταν κινούμαστε στο World Wide Web. Στο Web υπάρχουν πολλά προγράμματα περιήγησης για κάθε πλατφόρμα. Δημοφιλή προγράμματα περιήγησης είναι το Internet Explorer, το Mozilla Firefox και το Google Chrome. Οι παραπάνω εφαρμογές προσφέρονται δωρεάν. Ο βασικός σκοπός ενός προγράμματος περιήγησης είναι η σύνδεση με διακομιστές web, η αίτηση εγγράφων και η κατάλληλη μορφοποίηση τους. Κάθε ιστοσελίδα είναι ένα αρχείο, το οποίο έχει γραφτεί σε μια γλώσσα που ονομάζεται «γλώσσα σημείωσης υπερκειμένου» (HTML), ο οποία περιλαμβάνει το κείμενο της σελίδας, μια περιγραφή της δομής του και συνδέσεις με άλλα έγγραφα. Ο ρόλος του περιηγητή είναι να παίρνει τις πληροφορίες που λαμβάνει από τον διακομιστή Web και να τις μορφοποιεί αναλόγως. Διαφορετικά προγράμματα περιήγησης ενδέχεται να μορφοποιούν με διαφορετικούς τρόπους το ίδιο αρχείο.

2.2.1 Microsoft Internet Explorer

Δημιουργήθηκε από την Microsoft και περιλαμβάνεται στα Windows. Επειδή τα Windows κατέχουν μεγάλο μερίδιο στην αγορά λειτουργικών συστημάτων ο Internet Explorer είναι δημοφιλής περιηγητής. Πολλοί χρήστες διαδικτύου επιλέγουν να αντικαταστήσουν τον Internet Explorer με κάποιο άλλο πρόγραμμα περιήγησης λόγο προβλημάτων στην ασφάλεια και περισσότερων δυνατοτήτων που προσφέρουν άλλοι περιηγητές.

2.2.2 Mozila Firefox

Είναι ένα πρόγραμμα περιήγησης ανοιχτής πηγής, παρέχεται δωρεάν και κατέχει πάνω από το 30% της αγοράς των περιηγητών. Έχει πολλούς υποστηρικτές κυρίως επειδή δεν έχει προβλήματα με την ασφάλεια. Επίσης παρέχει ένα μεγάλο αριθμό επεκτάσεων που βελτιώνουν την εμπειρία στο Web. Ο Firefox έχει καταφέρει να συμβαδίζει με τα πρότυπα του Web τη στιγμή που εξελίσσονται. Διατίθεται για Windows, Linux και Mac OS X και μπορείτε να το κατεβάσετε δωρεάν από τη σελίδα www.mozilla.com

2.2.3 Google Chrome

Είναι το πιο φρέσκο προϊόν στην αγορά των περιηγητών. Χρησιμοποιεί τον ίδιο μηχανισμό ανοιχτής πηγής με το Safari. Είναι γνωστό για την υψηλή του απόδοση. Επιπλέον διαθέτει χαρακτηριστικά που αποτρέπουν τις συχνές καταρρεύσεις, ένα πρόβλημα που εμφανίζεται συχνά σε άλλα προγράμματα περιήγησης. Διατίθεται δωρεάν στη σελίδα www.google.com/intl/en/chrome/browser

2.3 Ιστοσελίδα

Οι ιστοσελίδες είναι ένα είδος εγγράφου του παγκόσμιου ιστού. Περιλαμβάνουν πληροφορίες με τη μορφή κειμένου, εικόνας, βίντεο και ήχου. Πολλές ιστοσελίδες μαζί συνθέτουν έναν ισότοπο. Οι σελίδες ενός ισότοπου εμφανίζονται κάτω από το ίδιο domain name, πχ teiser.gr. Ο χρήστης κάνοντας κλικ σε ένα σύνδεσμο μπορεί να μεταβεί σε κάποια άλλη σελίδα. Ο σύνδεσμος είναι είτε κείμενο είτε φωτογραφία και συνήθως για να φαίνεται ξεκάθαρα είναι υπογεγραμμένος με μπλε χρώμα. Γενικά η κατασκευή μιας ιστοσελίδας μπορεί να γίνει εύκολα με τη χρήση προγραμμάτων που κυκλοφορούν δωρεάν στο διαδίκτυο ή με τη χρήση προτύπων τα οποία όμως συνήθως έχουν κόστος.

Δυο είδη ιστοσελίδων είναι οι στατικές και οι δυναμικές και όσον αφορά την εμφάνιση τους δεν έχουν μεγάλες διαφορές μεταξύ τους.

2.3.1 Στατικές Ιστοσελίδες

Οι στατικές σελίδες είναι αυτές που το περιεχόμενο τους μεταφέρεται στον χρήστη όπως ακριβώς είναι αποθηκευμένο στον web server. Ο χρήστης δεν αλληλεπιδρά ουσιαστικά με τη σελίδα. Δε χρειάζονται προγραμματιστικές δεξιότητες ώστε να δημιουργήσει κανείς μια στατική ιστοσελίδα ούτε ειδικό λογισμικό στον εξυπηρετητή

ιστοσελίδων για τη δημοσίευση τους. Είναι έγγραφα τα οποία περιέχουν κείμενα, συνδέσμους και φωτογραφίες. Οι στατικές ιστοσελίδες είναι αποθηκευμένες συνήθως σε μορφή HTML και μεταφέρονται χρησιμοποιώντας το πρωτόκολλο HTTP

2.3.2 Δυναμικές Ιστοσελίδες

Οι δυναμικές σελίδες δεν είναι είδος εγγράφου, όπως οι στατικές, αλλά εφαρμογές! Δημιουργούνται δυναμικά τη στιγμή της πρόσβασης του χρήστη σε αυτές. Συνδέονται με Βάση Δεδομένων απ' όπου αντλούν το περιεχόμενο τους ανάλογα με το τι ζητάει ο χρήστης. Η χρήση των βάσεων δεδομένων είναι αυτή που δίνει τον χαρακτήρα δυναμική ιστοσελίδα, καθώς δε χρειάζεται κανείς να επεξεργαστεί το περιεχόμενο της ίδιας τις ιστοσελίδας κάθε φορά που είναι απαραίτητη αλλαγή, αλλά διαχειρίζεται έμμεσα το περιεχόμενο της μέσω της βάσης δεδομένων.

2.4 Http Πρωτόκολλο Επικοινωνίας

Ο browser και ο web server επικοινωνούν με ένα πρωτόκολλο, που ονομάζεται **HyperTextTransfer Protocol (HTTP).** Το HTTP παρέχει δυνατές λειτουργίες, πέρα από την απλή μεταφορά εγγράφων. Η πιο σημαντική λειτουργιά είναι η δυνατότητα να εκτελεί προγράμματα, με ορίσματα που παρέχονται από τον χρήστη και να παραδίδει τα αποτελέσματα ξανά ως html έγγραφα

2.5 Διακομιστές Web

Ο όρος διακομιστής Web μπορεί να αναφέρεται είτε σε software είτε σε hardware.Web Server ονομάζεται το πρόγραμμα που χρησιμοποιείται για την διανομή ιστοσελίδων αλλά και ο υπολογιστής στον οποίο εκτελείται αυτό το πρόγραμμα. Είναι η πύλη εισόδου ενός χρήστη προς το σύστημα και διαμεσολαβεί για την παροχή δεδομένων προς το user interface μιας web εφαρμογής. Μια ιστοσελίδα για να λειτουργήσει πρέπει να είναι αποθηκευμένη σε κάποιο server. Η πιο κοινή χρήση των web servers είναι να φιλοξενούν ιστοσελίδες, αλλά υπάρχουν και άλλες εφαρμογές όπως αποθήκευση δεδομένων και παιχνίδια.

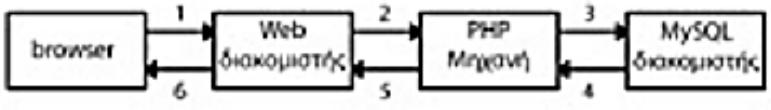
2.5.1 Apache Http Server

Ο Apache είναι ο πιο διαδεδομένος διακομιστής . Τον Οκτώβριο του 2007 κατείχε το 47,73% όλων των ισοτόπων. Ήταν η πρώτη εναλλακτική λύση μετά το Netscape Communications Corporation web server και αρχικά δημιουργήθηκε για χρήση σε Linux.

2.6 Επικοινωνία Client- Server

Εδώ περιγράφεται το τι γίνεται όταν ένας Browser κάνει μια αίτηση σε έναν διακομιστή ο οποίος παρέχει μια δυναμική σελίδα.

Όσον αφορά την παράδοση των πληροφοριών το ίντερνετ υιοθετεί το μοντέλο client-server. Ο **server** παρέχει υπηρεσίες σε έναν ή περισσότερους clients και μοιράζει επιλεκτικές πληροφορίες. Οι υπηρεσίες αυτές μπορεί να αφορούν την εύρεση πληροφοριών και την αποστολή τους στον client, όπως γίνεται στην περίπτωση των ερωτήσεων σε μια βάση δεδομένων στο web. Η διανομή των πόρων του server ονομάζεται time-sharing γιατί επιτρέπει πολλαπλές εφαρμογές να χρησιμοποιούν τους πόρους του υπολογιστή ταυτόχρονα. Ο **client** είναι πάντα ο αιτών των υπηρεσιών και δεν είναι τίποτα άλλο παρά ένας απλός υπολογιστής.



Συγκεκριμένα μια τυπική συναλλαγή σε μια Web βάση δεδομένων αποτελείται από τις παρακάτω φάσεις (δες εικόνα)

1. Ο Web Browser ενός χρήστη κάνει μια HTTP αίτηση για μια συγκεκριμένη ιστοσελίδα.
2. Ο Web διακομιστής λαμβάνει την αίτηση για τη σελίδα, ανακαλεί το αρχείο στο οποίο βρίσκεται και το περνά στην php μηχανή για επεξεργασία
3. Η php μηχανή αρχίζει την ανάλυση του κώδικα. Μέσα στον κώδικα υπάρχει μια εντολή η οποία κάνει τη σύνδεση με τη βάση δεδομένων και εκτελεί μια σειρά από ερωτήματα. Η php ανοίγει μια σύνδεση με τον MySQL διακομιστή και στέλνει το κατάλληλο ερώτημα
4. Ο MySQL διακομιστής λαμβάνει το ερώτημα της βάσης δεδομένων και αφού το επεξεργαστεί επιστρέφει τα αποτελέσματα
5. Η php μηχανή σταματά την εκτέλεση του κώδικα και επιστρέφει την τελική HTML σελίδα στον Web διακομιστή.
6. Ο Web διακομιστής περνά την HTML σελίδα ξανά στον Browser

Η http σύνδεση διαρκεί μόνο κατά τη διάρκεια της ανταλλαγής πληροφοριών.

Ουσιαστικά το μοντέλο επιτρέπει στον Web να θεωρείται ως ένα αποθηκευτικό μέσο και βάση δεδομένων απεριόριστης χωρητικότητας κατανεμημένη μεταξύ χιλιάδων υπολογιστών, οι οποίοι είναι προσβάσιμοι από οποιοδήποτε ανεξάρτητο pc.

2.7 Sessions και Cookies

Τα *Cookies* είναι μέρος των περιηγητών και επιτρέπουν σε μια σελίδα να ορίζει τιμές οι οποίες αποθηκεύονται από το πρόγραμμα περιήγησης και επιστέφουν στον διακομιστή κάθε φορά που ο χρήστης αιτείται μια σελίδα. Είναι χρήσιμα γιατί όταν ένας χρήστης συνδεθεί στη σελίδα, αποθηκεύονται στον υπολογιστή του cookies ώστε να παρακολουθούμε ποιος είναι και να μη χρειάζεται να τον υποχρεώνουμε να συνδέεται ξανά κάθε φορά που βλέπει μια σελίδα, η οποία προστατεύεται με κωδικό πρόσβασης. Άλλες πληροφορίες που δίνει ένα Cookie είναι η επισκεψιμότητα κάθε χρήστη.
Οι *σύνοδοι* δίνουν τη δυνατότητα να αποθηκεύονται δεδομένα μεταξύ των αιτήσεων στον διακομιστή. Γενικά τα sessions χρησιμοποιούνται μαζί με τα cookies προκειμένου ο διακομιστής να καταγράφει ποια σύνοδος σχετίζεται με ένα συγκεκριμένο χρήστη

2.8 HTML(Hypertext Markup Language)

Είναι η κύρια γλώσσα σήμανσης για τις ιστοσελίδες. Είναι μια client side γλώσσα. Η HTML δείχνει στον browser πώς να εμφανίσει το κείμενο. Ονομάζεται και γλώσσα σημειοθέτησης επειδή λειτουργεί εμπλουτίζοντας κανονικό κείμενο με «σημάδια» τα οποία έχουν ένα συγκεκριμένο νόημα για τον πλοηγό που χειρίζεται το έγγραφο. Οι εντολές ονομάζονται **ετικέτες** και αποτελούνται από ετικέτες αρχής και τέλους αντίστοιχα. Μια ιστοσελίδα αποτελείται από δυο βασικά μέρη: τη κεφαλίδα και το σώμα. Η **κεφαλίδα** περιέχει τον τίτλο της σελίδας και παραμέτρους που θα χρησιμοποιήσει το πρόγραμμα περιήγησης ιστοσελίδων. Στο **σώμα** αναλύεται το πραγματικό περιεχόμενο της σελίδας και περιλαμβάνει το κείμενο με ετικέτες. Τα έγγραφα της HTML έχουν τη δυνατότητα να περιέχουν ήχο, εικόνες και βίντεο.

2.9 Βάσεις Δεδομένων MySQL

Η MySQL είναι η πιο ευρέως διαδιδομένη σχεσιακή βάση δεδομένων ανοιχτού περιεχομένου. Ακλουθούν μερικά από τα πλεονεκτήματα της MySQL

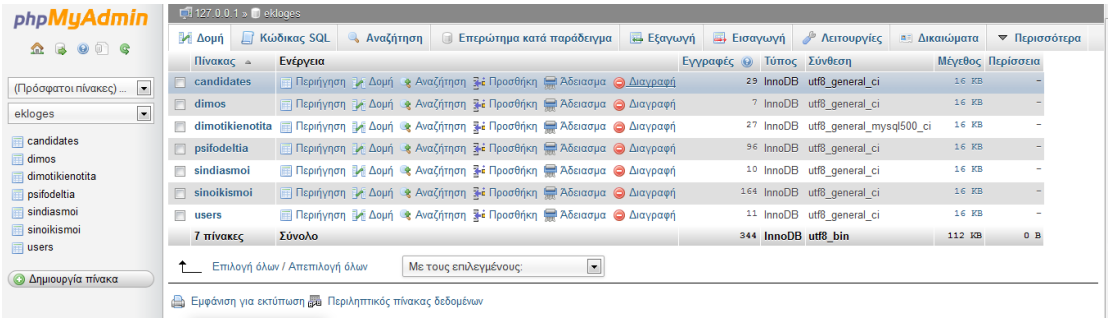
- Υψηλή Απόδοση (γρήγορη)
- Χαμηλό Κόστος (διατίθεται δωρεάν)
- Εύκολη διαμόρφωση και εκμάθηση
- Μεταφερσιμότητα (μπορεί να χρησιμοποιηθεί σε πολλά διαφορετικά συστήματα)
- Διαθεσιμότητα κώδικα προέλευσης

Στο παράδειγμα που ακολουθεί με τη χρήση εντολών MySQL συνδεόμαστε στη βάση δεδομένων.

```
$con = mysql_connect("localhost", "root","1234");
    if (!$con)
        die("unable to connect" . mysql_error());
echo 'Connected successfully';
```

Η εντολή mysql_connect παίρνει σαν όρισμα τον διακομιστή, το όνομα χρήστη και τον κωδικό και προσπαθεί να συνδεθεί με τη Βάση Δεδομένων. Σε περίπτωση που δε γίνει η σύνδεση εμφανίζεται mysql_error()"unable to connect" και τερματίζει τη προσπάθεια σύνδεσης.

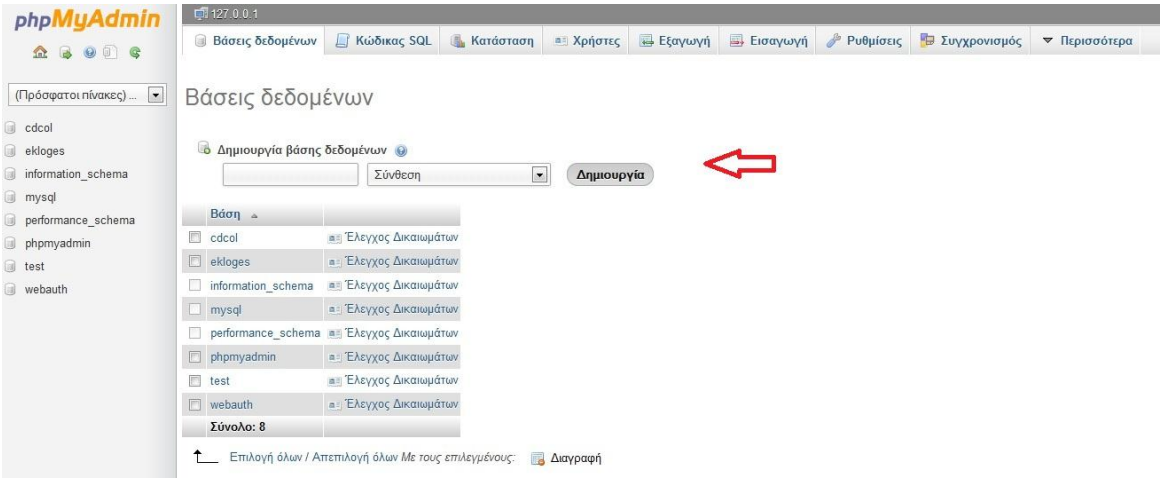
2.10.1 phpMyAdmin



Το phpMyAdmin είναι ένα δωρεάν εργαλείο ανοιχτού κώδικα γραμμένο σε php για τη διαχείριση βάσεων δεδομένων μέσω ενός browser. Είναι ένα χρήσιμο εργαλείο γιατί διευκολύνει την δημιουργία, την επεξεργασία και τη διαγραφή βάσεων δεδομένων, πινάκων, πεδίων και γραμμών εκτελώντας sql ερωτήματα με το πάτημα ενός κουμπιού.

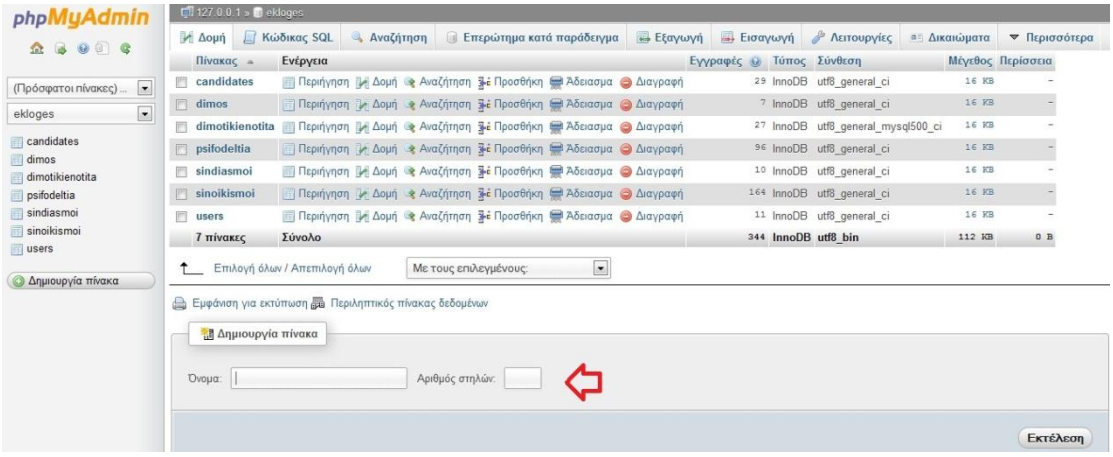
Δημιουργία Βάσης

Η δημιουργία μιας βάσης γίνεται από την αρχική σελίδα της εφαρμογής στη καρτέλα Databases. Ορίζεται το όνομα της βάσης δεδομένων και με το πάτημα του κουμπιού Create δημιουργείται η βάση δεδομένων.

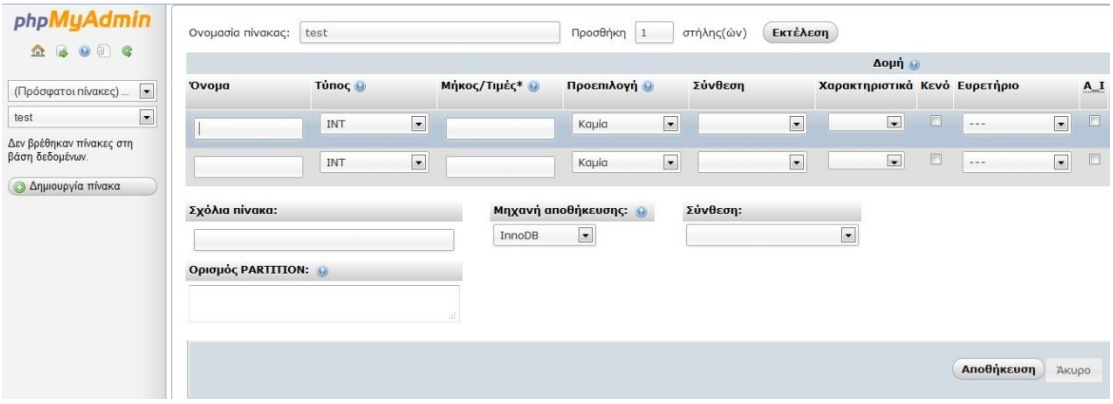


Δημιουργία Πίνακα

Επιλεγούμε το όνομα και το πλήθος των στηλών του πίνακα και πατάμε το κουμπί εκτέλεση



Εισάγουμε τα ονόματα των πεδίων, τους τύπους δεδομένων τη σύνθεση και το μήκος του πεδίου



2.11 PHP

Είναι μια ευρέως διαδεδομένη γλώσσα προγραμματισμού που χρησιμοποιείται για τη δημιουργία ιστοσελίδων με **δυναμικό περιεχόμενο**. Σε αντίθεση με την HTML είναι μια Server Side γλώσσα προγραμματισμού. Μια σελίδα PHP περνά από επεξεργασία από ένα συμβατό διακομιστή παγκοσμίου ιστού, πχ. Apache, ώστε να παραχθεί σε πραγματικό χρόνο το τελικό περιεχόμενο που θα σταλεί στο πρόγραμμα περιήγησης των επισκεπτών με τη μορφή κώδικα HTML.

Πλεονεκτήματα PHP

- Υψηλή απόδοση (γρήγορη)
- Διασυνδέσεις με διάφορα συστήματα Βάσεων Δεδομένων
- Ενσωματωμένες Βιβλιοθήκες για συνηθισμένες Web διαδικασίες (GD Βιβλιοθήκη)
- Χαμηλό κόστος
- Ευκολία εκμάθησης και χρήσης
- Μεταφερσιμότητα (είναι διαθέσιμη για πολλά λειτουργικά συστήματα)
- Διαθεσιμότητα Κώδικα προέλευσης

2.12 JavaScript

Το JavaScript είναι μια γλώσσα δημιουργίας δεσμών ενεργειών, η οποία μετατρέπει ιστοσελίδες σε εφαρμογές. Με τη βοήθεια του το πρόγραμμα περιήγησης γίνεται μια πλατφόρμα η οποία θα μπορούσε να εκτελέσει προγράμματα. Με το JavaScript τα προγράμματα αυτά περιλαμβάνονται ως τμήματα των ιστοσελίδων. Χρησιμοποιεί αυτό που ονομάζουμε *μοντέλο εκτέλεσης καθοδηγούμενο από συμβάντα*. Έτσι ο κώδικας JavaScript που ενσωματώνουμε σε μια σελίδα δεν εκτελείται μέχρι να ενεργοποιηθεί το συμβάν που σχετίζεται με τον κώδικα.

Μια απλή εφαρμογή JavaScript έχει την εξής δομή:

```
<html>
<head>
<title>This is a JavaScript example</title>
<script language="JavaScript">
<!--
document.write("Hello World!");
//-->
</script>
</head>
```

```
<body></body>
</html>
```

Το πρόγραμμα εμφανίζει το κείμενο “Hello World”. Με τη χρήση της εντολής document.write ορίζουμε ότι θέλουμε να εμφανιστεί με τη φόρτωση της σελίδας.

Όπως και τα Διαδοχικά φύλλα στυλ (CSS) το JavaScript μπορεί να ενσωματωθεί σε ιστοσελίδες με πολλούς τρόπους. Η σύνταξη της είναι επηρεασμένη από τη C. Χρησιμοποιείται και σε εφαρμογές εκτός ιστοσελίδων, όπως σε έγγραφα PDF.

Μαζί με το web επεκτάθηκε και αυτή και πλέον υποστηρίζεται από όλους τους περιηγητές και όλες τις web τεχνολογίες όπως Adobe Flash, AJAX, Microsoft Silverlight.

2.12.1Βιβλιοθήκες JavaScript (JavaScript frameworks)

Οι βιβλιοθήκες JavaScript αρχικά δημιουργήθηκαν για να λύσουν προβλήματα ασυμβατότητας μεταξύ των προγραμμάτων περιήγησης. Για παράδειγμα ορισμένα προγράμματα περιήγησης επιτρέπουν την ανάκτηση στοιχείων από το έγγραφο μέσω του ονόματος κλάσης και της μεθόδου *getElementByClassName()*. Εάν η δέσμη ενεργειών εξαρτάται από αυτή τη μέθοδο μπορεί, σε μερικά προγράμματα περιήγησης, να δημιουργηθεί πρόβλημα. Για να λυθεί αυτό, πρέπει να ελεγχθεί αν υπάρχει η μέθοδος πριν χρησιμοποιηθεί, ενώ αν δεν υπάρχει πρέπει να χρησιμοποιηθούν οι αντίστοιχες τεχνικές που είναι συμβατές με το πρόγραμμα περιήγησης. Όλες αυτές οι λύσεις συσκευάστηκαν σε πακέτα ώστε να δημιουργηθεί ένα απλούστερο περιβάλλον για κοινές λειτουργίες, οι οποίες λύνουν το πρόβλημα της ασυμβατότητας μεταξύ των προγραμμάτων περιήγησης. Έτσι διευκολύνουν την δημιουργία λογισμικού επιτρέποντας στον σχεδιαστή να αφιερώνει τον χρόνο του στον σωστό σχεδιασμό της εφαρμογής του παρά στο να ασχολείται με προγραμματιστικές λεπτομέρειες.

Γνωστά frameworks:

- jQuery
- Prototype
- Dojo
- ExtJs

2.12.1.1 jQuery

Είναι η πιο διαδεδομένη βιβλιοθήκη JavaScript αυτή τη στιγμή. Ο λόγος είναι ότι μαθαίνεται εύκολα και οι νέοι χρήστες μπορούν να πετύχουν πολλά με σχετικά λίγο κώδικα.

Επίσης αυτό που ξεχώρισε αρχικά τη jQuery είναι η δυνατότητα *χρήσης επιλογέων CSS* στο JavaScript. Η στοχοποίηση συγκεκριμένων στοιχείων σε μια σελίδα, ήταν

ένα δύσκολο πρόβλημα για τους προγραμματιστές του JavaScript και με τη χρήση του jQuery λύθηκε.
Μπορείτε να κατεβάσετε το jQuery και να μάθετε περισσότερα για αυτό στη διεύθυνση Http://jquery.com

2.12.1.2 PROTOTYPE

Η prototype έχασε έδαφος όταν εμφανίστηκε το jQuery, εξ αιτίας της ευκολίας που παρέχει στον χρήστη το δεύτερο. Είναι διάσημο χάρη στο πρότυπο *Scriptaculous* που χρησιμοποιεί. Το Scriptaculous είναι μια συλλογή κινήσεων που παρέχει στα στοιχειά της σελίδας δυνατότητες όπως να συρρικνώνονται, να εξαφανίζονται σταδιακά και άλλα εφέ. Παρόμοιες βιβλιοθήκες εφέ έχουν εισαχτεί σε άλλες βιβλιοθήκες αλλά το Scriptaculous ήταν αυτό που έδειξε τις δυνατότητες των JavaScript και των CSS για τη δημιουργία οπτικών εφέ.
Μπορείτε να κατεβάσετε το Prototype και το Scriptaculous και να μάθετε περισσότερα για αυτά στη διεύθυνση http://www.prototypejs.org και http://script.aculo.us/ αντίστοιχα.

2.12.1.3 Dojo

Η Dojo έχει υιοθετηθεί ευρέως από εταιρικά site και χρησιμοποιείται σε έτοιμες εφαρμογές Web. Περιλαμβάνει περισσότερες λειτουργίες από το Jquery, όπως λειτουργίες πλέγματος δεδομένων, με τις οποίες δημιουργούνται πίνακες με εύκολη ταξινόμηση. Η βιβλιοθήκη που χρησιμοποιεί το Dojo ονομάζεται *Dijit*.
Μπορείτε να κατεβάσετε το Dojo και να μάθετε περισσότερα για αυτό στη διεύθυνση www.dojotoolkit.org

2.12.1.4 EXTJS

Δημιουργήθηκε και αναπτύχτηκε από την εταιρία Sencha. Χρησιμοποιεί τεχνικές όπως **Ajax**, **DHTML** και **Dom** και έχει σχεδιαστεί έτσι ώστε να συνεργάζεται με άλλα συστήματα όπως **jQuery** και **Prototype.** Αρχικά είχε χρησιμοποιηθεί ως πρόσθετο για την **YUI** βιβλιοθήκη της Yahoo. Από τη πρώτη έκδοση 1,1 λειτούργει αυτόνομα κάνοντας τη χρήση εξωτερικών βιβλιοθηκών προαιρετική. Περιλαμβάνει σετ από widgets για χρήση εντός των δικτυακών εφαρμογών όπως textfield, datafields, listbox, combobox, grids, trees, toolbars

Γενικότερα κάνει τη δουλειά του προγραμματιστή πολύ πιο εύκολη και το αποτέλεσμα όμορφο και πλούσιο. Σε ορισμένες όμως περιπτώσεις είναι δύσκολη η

αποσφαλμάτωση του προγράμματος, ειδικά για αρχάριο χρήστη. Με τη βοήθεια του FireBug(εργαλείο παρακολούθησης του περιηγητή firefox) παρατήρησα πως πολλές φορές εμφανίζονται μηνύματα σφάλματος σε αρχεία του προγράμματος, ενώ το λάθος βρίσκονταν σε άλλο αρχείο, κάνοντας έτσι δύσκολο τον εντοπισμό του.

Επιλέχτηκε χάρη στη πληθώρα παραδειγμάτων που βρίσκονται στη σελίδα της sencha (http://www.sencha.com/products/extjs/examples/) και χάρη στο documentation το οποίο διευκολύνει την χρήση των συναρτήσεων του ext (http://docsncha.se.com/extjs/2.3.0/#!/api)
Η έκδοση που χρησιμοποιήθηκε για τη δημιουργία της εφαρμογής είναι η 2.3.0 .
Μπορείτε να κατεβάσετε το ExtJs και να μάθετε περισσότερα για αυτό στη διεύθυνση http://www.sencha.com/products/extjs/

2.12.1.4.1 Εγκατάσταση του ExtJs

Η εγκατάσταση το ext δεν είναι ιδιαίτερα δύσκολη. Αφού κατεβάσουμε το πρόγραμμα κάνουμε extract τα αρχεία και τα τοποθετούμε στο htdocs μαζί με τα υπόλοιπα αρχεία της εφαρμογής

2.13 Xampp

ο xampp είναι μια δωρεάν πλατφόρμα η οποία περιέχει τις τελευταίες εκδόσεις από

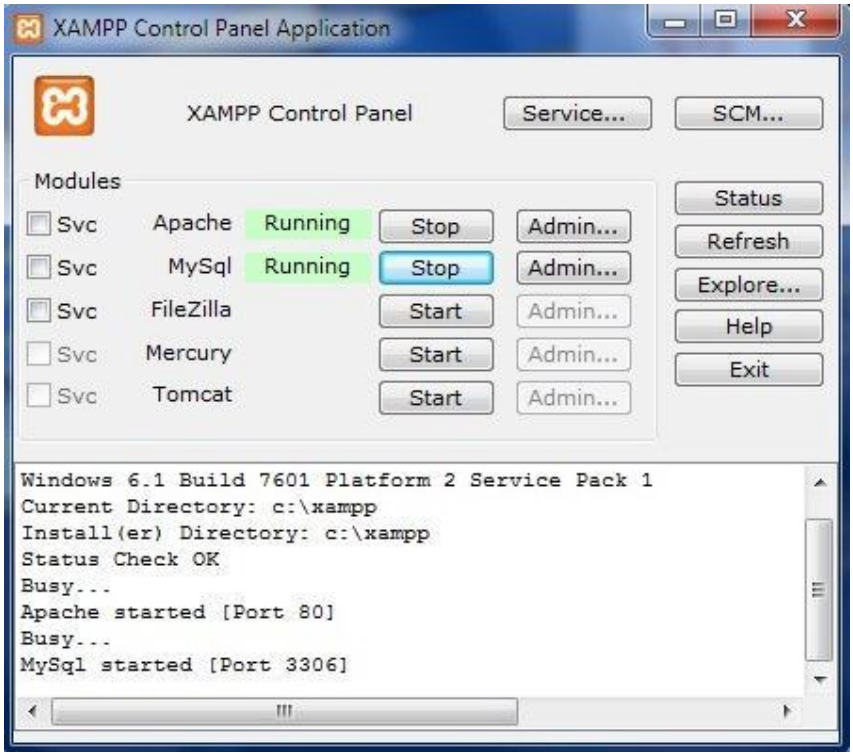
- Apache HTTP Server
- MySQL database
- Μεταγλωττιστές για script που γράφηκαν σε php ή Pearl γλώσσα.

Με την εγκατάσταση του δημιουργείται ένας τοπικός εξυπηρετητής για τα αρχεία που βρίσκονται στον φάκελο ***localhost***. Για τη διαχείριση της βάσης δεδομένων χρησιμοποιείται το εργαλείο *phpMyAdmin*. Το xampp είναι εύκολο στην εγκατάσταση. Μπορείτε να το κατεβάσετε στη διεύθυνση http://www.apachefriends.org/en/xampp-windows.html

2.13.1Εγκατάσταση Xampp

Η εγκατάσταση γίνεται τοπικά οπότε πρέπει να χρησιμοποιηθεί ένας τοπικός εξυπηρετητής πάνω στον οποίο θα στηθεί η εφαρμογή.

Αφού τελειώσει η εγκατάσταση του Xampp τρέχουμε το πρόγραμμα από τη συντόμευση και πατάμε Start στον Apache και στη MySQL.



Πληκτρολογούμε Http://127.0.0.1 ή Http://localhost/ για να μεταβούμε στη κεντρική σελίδα του XAMPP από όπου μπορούμε να χρησιμοποιήσουμε τα εργαλεία που έχει στο πακέτο όπως το PhpMyAdmin.

2.14 NetBeans

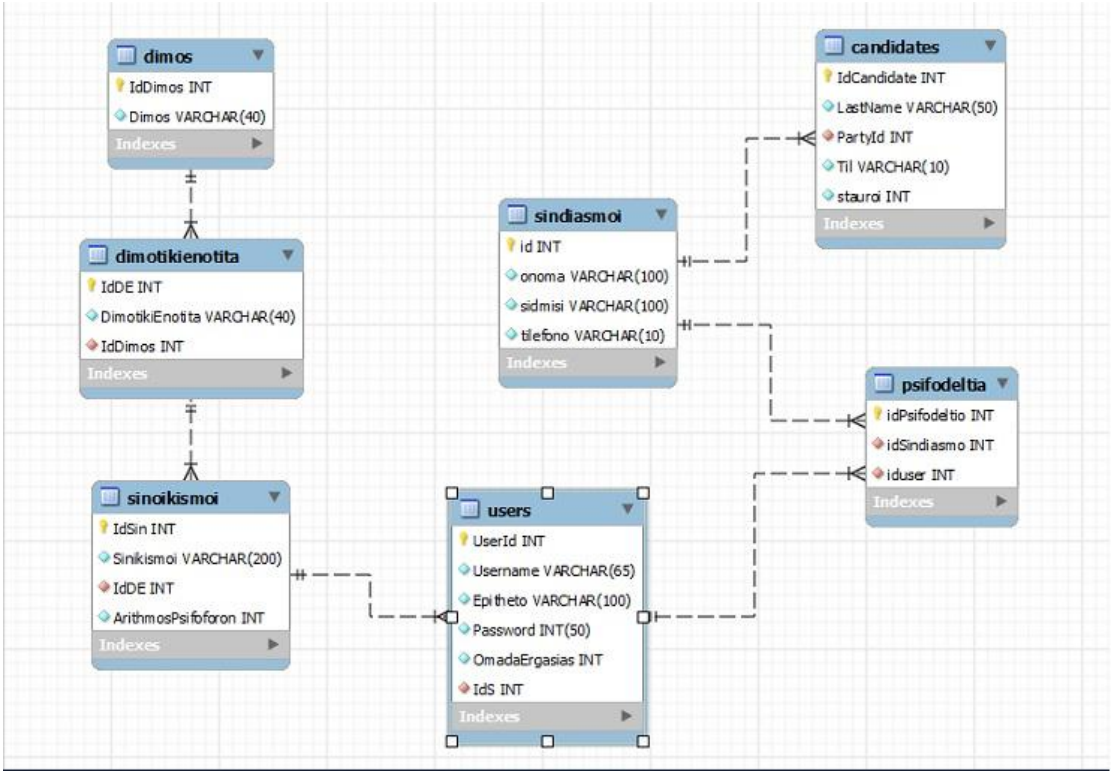
Το NetBeans είναι μια πλατφόρμα για την δημιουργία εφαρμογών. Είναι γραμμένο σε Java και υπάρχουν εκδόσεις για Windows, OsX, Linux και άλλες πλατφόρμες. Επιλέχθηκε για τις πλούσιες λειτουργιές του που το ξεχωρίζουν από τους άλλους Editors. Μπορεί να κάνει αποσφαλμάτωση κώδικα. Διαθέτει λειτουργία ιστορικού επεξεργασίας κώδικα ώστε να μπορεί ο χρήστης εύκολα να δει τις τελευταίες αλλαγές που έχει κάνει.

Μπορείτε να κατεβάσετε το NetBeans στη διεύθυνση https://netbeans.org/downloads

Κεφάλαιο 3: Ανάλυση της εφαρμογής

3.1 Σχεδίαση

3.1.1Διαγραμμα EER



Ο πίνακας ‘dimos’ αποτελείται από το πεδίο idDimos(int), κ.κ του πίνακα, και το πεδίο Dimos(varchar 40)

Ο πίνακας ‘dimotikienotita’ αποτελείται από το πεδίο idDE (int) κ.κ του πίνακα, το πεδίο DimotikiEnotita(VARCHAR 40) το οποίο περιέχει τις δημοτικές ενότητες και το πεδίο IdDimos(INT) ξένο κλειδί από τον πίνακα dimos

Ο πίνακας ‘sinoikismoi’ αποτελείται από το πεδίο idSin (int) κ.κ του πίνακα, το πεδίο Sinikismoi(VARCHAR 40) το οποίο περιέχει τους Συνοικισμούς, το πεδίο IdDE(INT) ξένο κλειδί από τον πίνακα dimotikienotita και το πεδίο ArithmosPsifoforon(int 100) που περιέχει τους εγγεγραμμένους ψηφοφόρους κάθε συνοικισμού.

Ο πίνακας ‘sindiasmoi’ αποτελείται από το πεδίο id (int) κ.κ του πίνακα, το πεδίο onoma(VARCHAR 100) το οποίο περιέχει το πλήρες όνομα των συνδυασμών, το πεδίο sidmisi το οποίο περιέχει τη συντόμευση του συνδυασμού για χάρη ευκολίας, σε ορισμένες περιπτώσεις που το όνομα είναι πολύ μεγάλο και το πεδίο tilefono(VARCHAR 100) το οποίο περιέχει τα το τηλέφωνο του κάθε συνδυασμού.

Ο πίνακας ‘candidates’ αποτελείται από το πεδίο idCandidate (int 255) κ.κ του πίνακα, το LastName(VARCHAR 50) το οποίο περιέχει το ονοματεπώνυμο, το πεδίο PartyId(int 11) ξένο κλειδί από τον πίνακα sindiasmoi, το πεδίο Til, το οποίο περιέχει το τηλέφωνο του υποψήφιου και το πεδίο stauroi στο οποίο καταχωρούνται αυξητικά οι σταυροί που πήρε ο κάθε υποψήφιος από την εκλογική αναμέτρηση

Ο πίνακας ‘users’ αποτελείται από το πεδίο UserId(int) κ.κ του πίνακα, το πεδίο Username(varchar65) το οποίο περιέχει το username του χρηστή για την σύνδεση του στο σύστημα, το πεδίο epitheto(varchar100), το οποίο περιέχει το ονοματεπώνυμο του χρήστη, το πεδίο password(int 50) με τον κωδικό του, το πεδίο OmadaErgasias(int) το οποίο μπορεί να είναι μόνο 0 ή 1 ανάλογα με τα δικαιώματα του χρήστη και το πεδίο IdS(int) ξένο κλειδί με το id του συνοικισμού στον οποίο καταχωρεί ο χρήστης.

Ο πίνακας ‘psifodeltia’ αποτελείται από το πεδίο idPsifodeltio(int11) κ.κ του πίνακα, το πεδίο idSindiasmo(int11)ξενο κλειδί από τον πίνακα sindiasmoi στο οποίο καταχωρείται το id του συνδυασμού του ψηφοδελτίου και τέλος το πεδίο iduser(int) ξένο κλειδί του πίνακα users στο οποίο καταχωρείται το id του χρήστη που κάνει τη καταχώρηση

3.2 Απαιτήσεις Περιβάλλοντος χρηστών

Ο Administrator:

Είναι υπεύθυνος για τη συμπλήρωση των δεδομένων που είναι απαραίτητα για τη σωστή λειτουργία της εφαρμογής.

Το σύστημα μπορεί σε κάθε εκλογική αναμέτρηση να αλλάζει πλήρως. Μετά από σύνδεση στο σύστημα ο administrator μπορεί να προσθέτει, διαγράφει και να τροποποιεί Δήμους, Δημοτικές Ενότητες, Εκλογικά διαμερίσματα σε περίπτωση που αυτά αλλάξουν, όπως έγινε πρόσφατα με το σχέδιο Καλλικράτης. Με αυτόν τον τρόπο η εφαρμογή μπορεί να τροποποιηθεί ώστε να αφορά άλλους Δήμους, πέρα από τις Σέρρες.

Ακόμα ο Administrator εισάγει Συνδυασμούς και τους υποψήφιους που ανήκουν σε αυτούς για κάθε εκλογική αναμέτρηση.

Τέλος είναι αυτός που προσθέτει τους χρήστες της εφαρμογής που θα έχουν είτε δικαιώματα administrator είτε, ως μέλη της εφορευτικής επιτροπής, θα καταχωρούν τα ψηφοδέλτια . Η υλοποίηση αυτού του τμήματος της εφαρμογής έγινε με το extJS

Ο καταχωρητής:

Ο χρήστης πρώτα πρέπει να ταχτοποιήσει τα στοιχειά του μέσω Login Φόρμας. Μετά το κλείσιμο των κάλπεων και μαζί με τον προεδρεύον υπάλληλο της εφορευτικής επιτροπής θα εισάγει κάθε ψηφοδέλτιο απευθείας στη βάση δεδομένων αντί σε χαρτί. Έτσι ελαχιστοποιείται ο χρόνος ταξινόμησης και υπολογισμού των

συγκεντρωτικών αποτελεσμάτων από ανθρώπινο παράγοντα και γίνεται πιο εύκολος ο υπολογισμός των αποτελεσμάτων. Η υλοποίηση αυτού του τμήματος της εφαρμογής έγινε με τη χρήση της PHP και της MySQL.

Ο χρήστης διαδικτύου

Όποιος χρήστης του διαδικτύου επιθυμεί μπορεί να δει τα αποτελέσματα τις διαδικασίας. Εμφανίζονται αποτελέσματα, ανά δήμο, δημοτική ενότητα και συνοικισμό ανάλογα με το τι πληροφορία επιθυμεί ο χρήστης.

Συγκεκριμένα εμφανίζονται τα συγκεντρωτικά ποσοστά των ψήφων που έλαβε ο κάθε συνδυασμός με τη χρήση γραφημάτων και πινάκων αλλά και συγκεντρωτικά αποτελέσματα όπως εγγεγραμμένοι ψηφοφόροι, αποχή, άκυρα και λευκά.

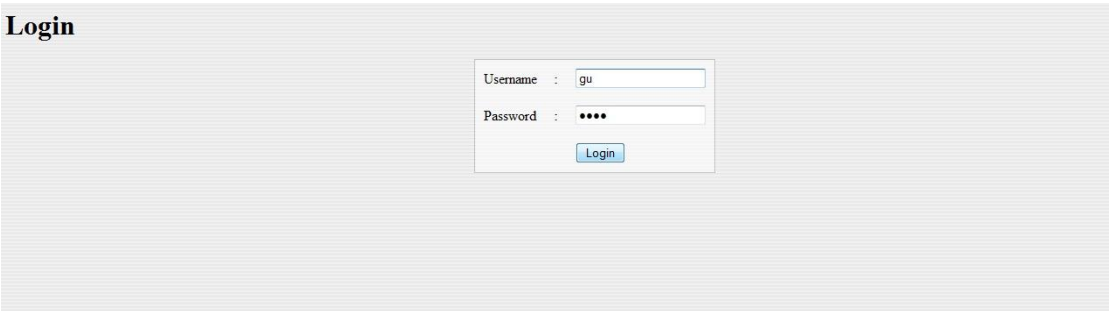
Τέλος εμφανίζονται οι υποψήφιοι που συγκέντρωσαν το μεγαλύτερο ποσοστό των ψήφων. Η υλοποίηση αυτής της εφαρμογής έγινε με τη βοήθεια της **GD** βιβλιοθήκης της PHP αλλά και της MySQL

Παρόμοια σελίδα έχει το υπουργείο εσωτερικών (http://ekloges.ypes.gr) της εταιρίας **Singular Logic** χωρίς όμως να δείχνει αναλυτικά τις ψήφους που πήρε ο κάθε υποψήφιος.

3.3 Σύνδεση στο σύστημα

Ο χρήστης για να εισέλθει στο περιβάλλον του Administrator ή του καταχωρητή χρειάζεται να συνδεθεί στο σύστημα

Γίνεται έλεγχος των στοιχείων του και ανάλογα με τα δικαιώματα που διαθέτει ο χρήστης μεταφέρεται στο Administator τμήμα της εφαρμογής ή στο τμήμα καταχώρησης των ψηφοδελτίων.

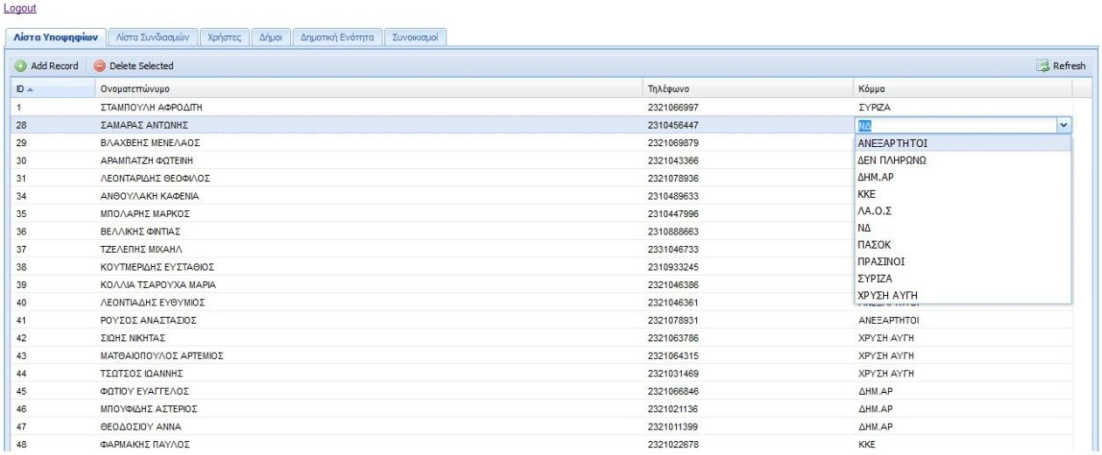


3.4 Επισκόπηση Περιβάλλοντος Διαχειριστή

3.4.1Παρουσίαση περιβάλλοντος

Αφού ο διαχειριστής εισέλθει στο σύστημα εμφανίζεται μια φόρμα με τις καρτέλες: Λίστα Υποψηφίων, Λίστα Συνδυασμών, Χρηστές, Δήμοι, Δημοτική Ενότητα, Συνοικισμοί. Στη κάτω μπάρα του grid εμφανίζονται πληροφορίες όπως το σύνολο των εγγραφών και κουμπιά paging σε περίπτωση που οι εγγραφές δε χωράνε σε μια καρτέλα. Μπορεί να προσθέσει εγγραφές πατώντας το κουμπί add Record, να διαγράψει με το κουμπί Delete Record, να κάνει ανανέωση στο grid με το κουμπί refresh, να ταξινομήσει τις εγγραφές και να επιλέξει ποια πεδία θέλει να εμφανίζονται. Όπου κρίνεται απαραίτητο η επιλογή των τιμών ενός πεδίου γίνεται με τη βοήθεια DropDown επιλογών. Έτσι γλιτώνεται χρόνος και ελαχιστοποιείται η πιθανότητα ορθογραφικών λαθών τα οποία θα οδηγούσαν σε λάθος λειτουργία της εφαρμογής.

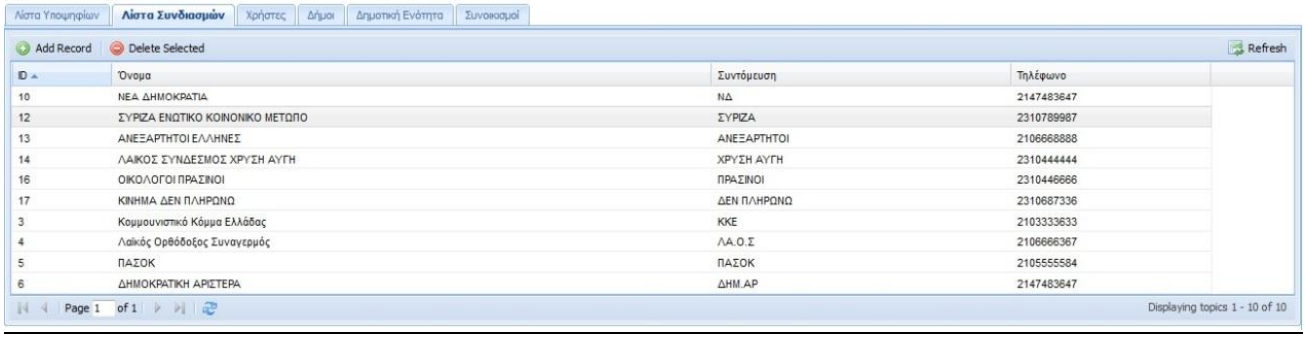
Καρτέλα Υποψηφίων:



Αποτελείται από τα πεδία :

1. ID, κύριο κλειδί του πίνακα το οποίο έχει αυτόματη αρίθμηση
2. Ονοματεπώνυμο, περιέχει το ονοματεπώνυμο του υποψήφιου
3. Τηλέφωνο, επιτρέπονται μόνο αριθμοί και το μέγεθος του μπορεί να είναι από 10 μέχρι 14 ψηφία
4. Κόμμα, περιέχει με τη μορφή Dropdown το πεδίο sidmisi του πίνακα sindiasmoi

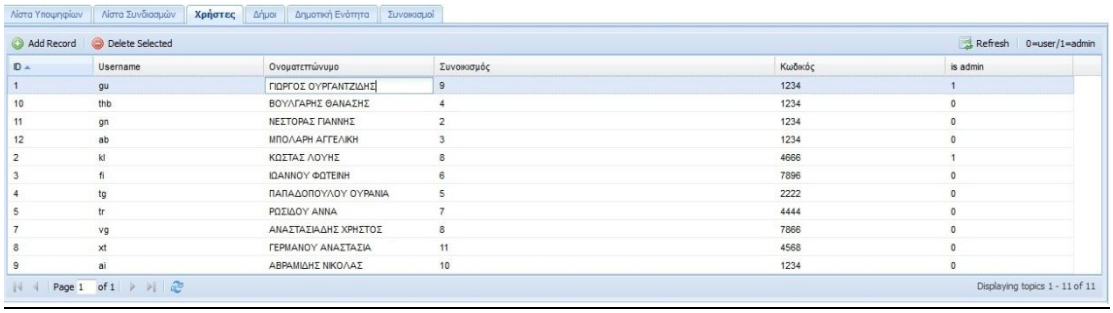
Καρτέλα Συνδυασμών



Αποτελείται από τα πεδία:

1. ID, κύριο κλειδί του πίνακα το οποίο έχει αυτόματη αρίθμηση
2. Όνομα, περιέχει ολόκληρο το όνομα του συνδυασμού
3. Συντόμευση, περιέχει τη συντόμευση του συνδυασμού σε περίπτωση που το όνομα είναι πολύ μεγάλο
4. Τηλέφωνο, επιτρέπονται μόνο αριθμοί και το μέγεθος του μπορεί να είναι από 10 μέχρι 14 ψηφία

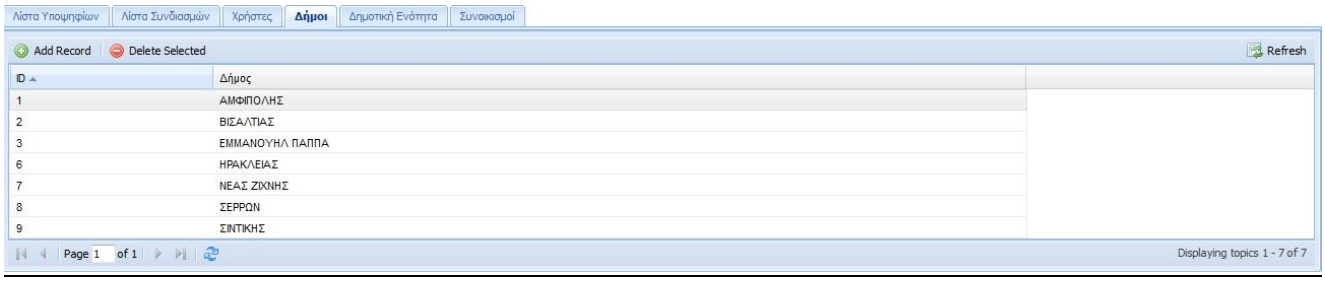
Καρτέλα Χρηστών



Αποτελείται από τα πεδία:

1. ID, κύριο κλειδί του πίνακα το οποίο έχει αυτόματη αρίθμηση
2. UserName, περιέχει το username με το οποίο θα συνδεθεί ο χρήστης
3. Όνομα, περιέχει ολόκληρο το όνομα του χρήστη
4. Συνοικισμός, περιέχει τον κωδικό του συνοικισμού στον οποίο θα καταχωρεί ο χρήστης τα ψηφοδέλτια
5. Το πεδίο κωδικός
6. Το πεδίο is Admin με τιμές 0 για απλό χρήστη της εφαρμογής, ο οποίος έχει πρόσβαση μόνο στο κομμάτι καταχώρησης ψηφοδελτίων και 1 για τους Administrator

Καρτέλα Δήμων:

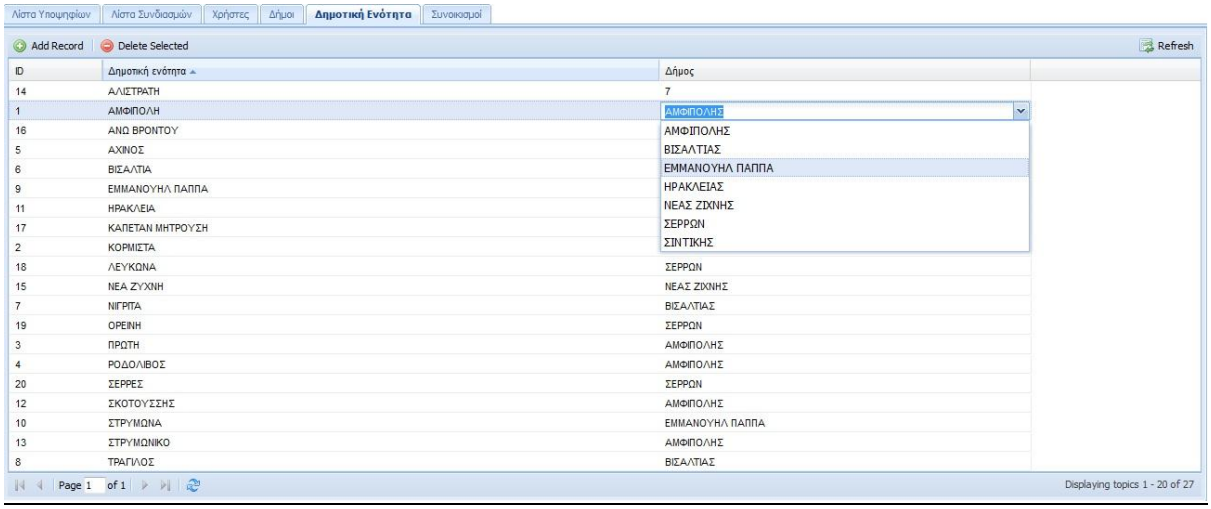


Αποτελείται από τα πεδία:

1. ID, κύριο κλειδί του πίνακα το οποίο έχει αυτόματη αρίθμηση
2. Δήμος, περιέχει τους Δήμους

Οι επιλογές Add Record και Delete Record είναι ενεργοποιημένες μόνο για τη περίπτωση που γίνει κάποια αλλαγή στη δόμηση του νομού Σερρών

Καρτέλα Δημοτικών Ενοτήτων :

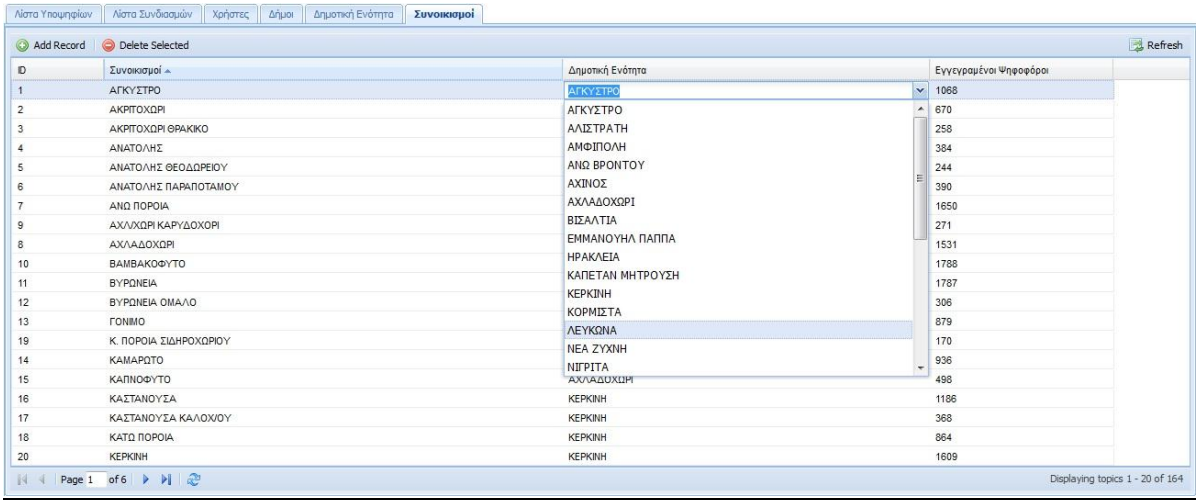


Αποτελείται από τα πεδία:

1. ID, κύριο κλειδί του πίνακα το οποίο έχει αυτόματη αρίθμηση
2. Δημοτική Ενότητα, περιέχει το όνομα των ενοτήτων
3. Δήμος, περιέχει με τη μορφή Dropdown το πεδίο id των δήμων και εμφανίζεται το πεδίο Dimos

Οι επιλογές Add Record και Delete Record είναι ενεργοποιημένες μόνο για τη περίπτωση που γίνει κάποια αλλαγή στη δόμηση του νομού Σερρών

Καρτέλα συνοικισμών :



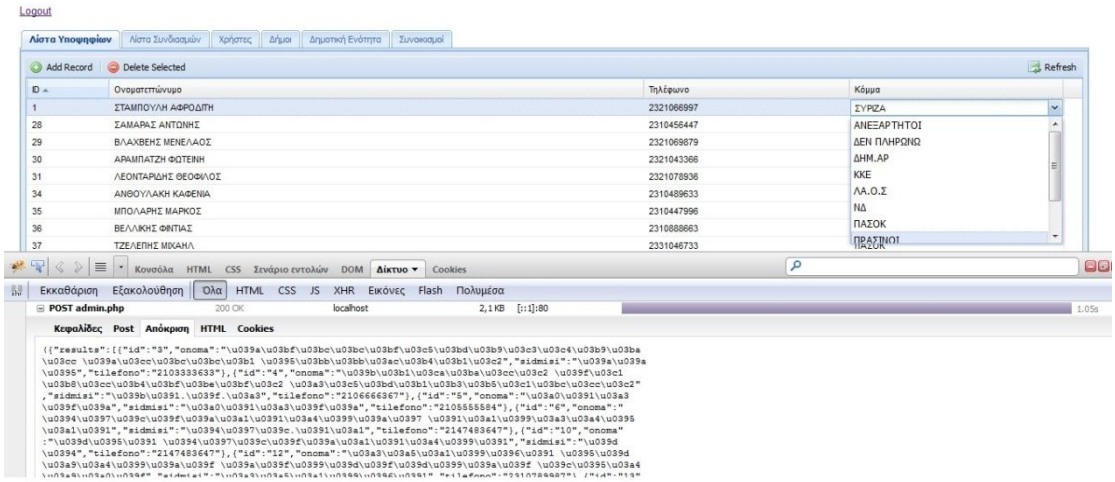
Αποτελείται από τα πεδία:

1. ID, κύριο κλειδί του πίνακα το οποίο έχει αυτόματη αρίθμηση
2. Συνοικισμοί, περιέχει το όνομα των συνοικισμών
3. Δημοτική Ενότητα, περιέχει με τη μορφή Dropdown το πεδίο id των δημοτικών Ενοτήτων και εμφανίζεται το πεδίο DimotikiEnotita
4. Εγγεγραμμένοι Ψηφοφόροι, αριθμητικό πεδίο που περιέχει το σύνολο των ψηφοφόρων του συνοικισμού

Οι επιλογές Add Record και Delete Record είναι ενεργοποιημένες μόνο για τη περίπτωση που γίνει κάποια αλλαγή στη δόμηση του νομού Σερρών

3.4.2 Επικοινωνία διακομιστή με την Βάση Δεδομένων (το στοιχείο FireBug)

Στην ενότητα αυτή θα αναλύσουμε τι συμβαίνει στο παρασκήνιο με τη βοήθεια του εργαλείου firebug. Το παράδειγμα που ακολουθεί δείχνει τι συμβαίνει όταν στη καρτέλα υποψήφιοι, κάνοντας κλικ στο πεδίο κόμμα ζητάμε να εμφανιστεί η λίστα με τους υπάρχοντες συνδυασμούς της Βάσης Δεδομένων.



Όπως βλέπουμε στην εικόνα, γίνεται μια κλήση με τη μέθοδο post στο αρχείο admin.php. Στη καρτέλα Post εμφανίζεται το τι ζητάμε από τον διακομιστή. Βλέπουμε ότι το πεδίο query task έχει τη τιμή readPartyId. Δηλαδή ζητάμε από τον διακομιστή να διαβάσει το id του συνδυασμού που ανήκει ο υποψήφιος. Στη καρτέλα απόκριση εμφανίζονται κωδικοποιημένες οι πληροφορίες που επιστρέφει ο διακομιστής.

3.4.3 Υλοποίηση Περιβάλλοντος Διαχειριστή(Γενική Δομή μιας ExtJs Εφαρμογής)

Όπως έχω αναφέρει παραπάνω η υλοποίηση της εφαρμογής του administrator έγινε με τη βοήθεια το ext 2.3.0. Για την ανάπτυξη της εφαρμογής χρησιμοποιήθηκε ως πρότυπο μια έτοιμη εφαρμογή του Michael Lecomte την οποία μπορείτε να βρείτε στη διεύθυνση: http://www.sencha.com/forum/showthread.php?18435-EditorGrid-working-example-%28php-mysql-backend%29

Η βασική δομή της εφαρμογής έχει ως εξής.

1.1 Δημιουργία Data Record

Δημιουργεί ένα κονστράκτορα για μια συγκεκριμένη διάταξη δεδομένων για τον καθορισμό των πεδίων που συνθέτουν μια γραμμή δεδομένων. Στιγμιότυπα αυτής της κλάσης περιέχουν πληροφορίες για τον καθορισμό ενός Record(γραμμή δεδομένων) αλλά και πληροφορίες για το περιεχόμενο του Record για χρήση σε Ext.Data.Store Αντικείμενα.

1.2 Καθορισμός του Reader

Ανήκει στη κλάση Ext.data.DataReader και ο ρόλος του είναι να συνδέει ένα Data αντικείμενο με το Data Store. Παίρνει την τιμή ενός πεδίου και δημιουργεί ένα Array από Ext.Data.Record αντικείμενα. Τα δεδομένα μπορεί να είναι σε πολλά διαφορετικά formats, οπότε πρέπει να χρησιμοποιήσουμε τον σωστό Reader, ο οποίος

να ξέρει να τα χειρίζεται. Δηλώνουμε Reader με βάση το format των δεδομένων που επιστρέφουμε (Xml, Json, Array) Για να διαμορφώσουμε έναν Reader πρέπει να του ορίσουμε που πρέπει να ψάξει για να βρει την απόκριση των δεδομένων, δηλαδή στο 'root' .Επίσης καθορίζουμε με το id το πρωτεύον κλειδί και με το total property τον συνολικό αριθμό των σειρών.

1.3 Δημιουργία Data Store(s)

Το Store αντιπροσωπεύει από τη μεριά του χρήστη τα δεδομένα του Server. Καθορίζουμε το

α. proxy (πως να διαβάσει τα δεδομένα, που να τα διαβάσει (url) και με ποια μέθοδο (get, post))

β. reader (παίρνει τα δεδομένα και τα σπάει σε στήλες)

2.1 Δημιουργία Column Model

Εδώ αποφασίζουμε ποια μέρη από τα δεδομένα (αυτά που ήρθαν από τον server) θέλουμε να εμφανίσουμε και πως θα τα εμφανίσουμε. Δηλαδή δηλώνουμε τι θα εμφανιστεί στο grid, και η σειρά που τα δηλώνουμε είναι αυτή που θα δει και ο χρήστης.

3.1 δημιουργία handelers για events (addRecord, Delete Record) κ.α.

Εδώ λέμε στο πρόγραμμα πώς να χειριστεί events όπως το edit και το delete μιας σειράς, την ανανέωση του grid, και την αποστολή εγγραφών στη βάση.

3.2 δημιουργία του grid

Δημιουργούμε το grid και αποφασίζουμε που, πως και πότε θέλουμε να εμφανιστεί.

3.3 προσθήκη Listeners στο grid

Δημιουργούμε Events. Συγκεκριμένα λέμε στο grid μετά από edit να καλέσει τη συνάρτηση που δημιουργήσαμε στο βήμα 3.1 και χειρίζεται το edit των πεδίων.

4.0 **render the grid**

Αυτό το βήμα μπορεί να παραληφθεί αν δηλώσουμε την ιδιότητα ‘renderTo’ στον κώδικα που δημιουργεί το grid. Ουσιαστικά δηλώνουμε στο grid σε ποιο να εμφανιστεί.

5.0 **φόρτωση των δεδομένων**

Σε αυτό το τελικό στάδιο φορτώνουμε τα Data Stores που δημιουργήσαμε στο βήμα 1.3

3.4.4 Κώδικας Υλοποίησης Περιβάλλοντος Διαχειριστή

Ο κώδικας υλοποίησης της εφαρμογής βρίσκεται στο παράρτημα και αποτελείται από τα αρχεία

- Administrator.php
- Admin.js
- Admin.php
- Common.php

Administrator.php

Το αρχείο που περιέχει την HTML Μορφοποίηση και τα CSS

Admin.php

Είναι το αρχείο που χειρίζεται τις κλήσεις στη Βάση Δεδομένων.

Ανάλογα με το τι επιθυμεί ο χρήστης καλείται η αντίστοιχη συνάρτηση.

Εδώ δηλαδή βρίσκεται η συνάρτηση readPartyId που είδαμε στο παράδειγμα με τις dropdown επιλογές των Συνδυασμών.

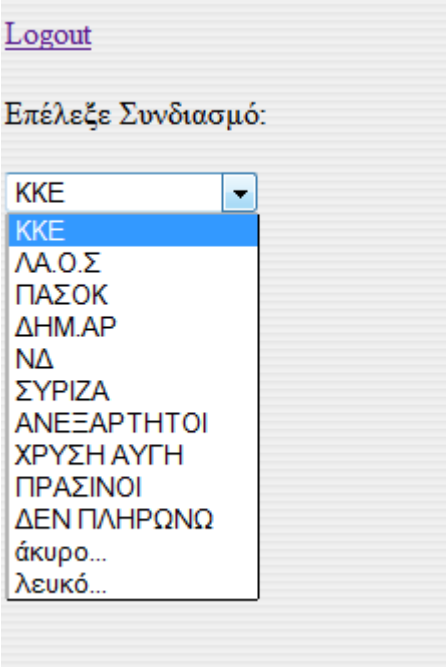
Admin.js

Ο πυρήνας της εφαρμογής. Έχει όλες τις συναρτήσεις για τη δημιουργία της εφαρμογής, όπως το setupDatasource και ο καθορισμός του Reader.

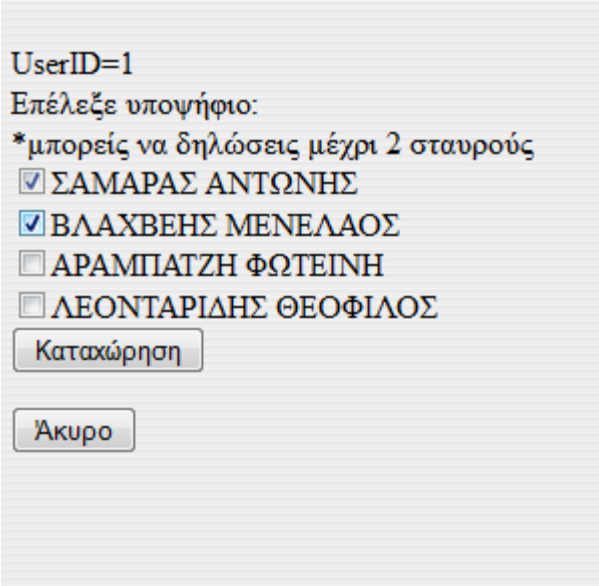
3.5 Επισκόπηση Περιβάλλοντος Καταχωρητή

3.5.1 Παρουσίαση περιβάλλοντος

Είναι ένα απλό περιβάλλον που δημιουργήθηκε με τη χρήση php και html.



Ο Χρήστης αφού εισέλθει στο σύστημα βλέπει την παραπάνω φόρμα στην οποία εμφανίζονται δυναμικά η λίστα με τους συνδυασμούς της βάσης Δεδομένων, και επιλέγει μεταξύ συνδυασμών, άκυρου και λευκού και μετά πατάει καταχώρηση. Σε περίπτωση που επιλέξει κάποιον από τους συνδυασμούς και όχι άκυρο ή λευκό του εμφανίζεται η φόρμα που φαίνεται στη παρακάτω εικόνα.



Εδώ είναι η φόρμα που επιλέγει τους σταυρούς. Μπορεί να δηλώσει από κανένα έως δυο checkboxes. Σε περίπτωση που επιλέξει παραπάνω του εμφανίζεται μήνυμα λάθους και ζητάει από τον χρήστη να ξανακάνει τη καταχώρηση. Ακόμα μπορεί με το κουμπί άκυρο να επιστρέψει στη προηγούμενη φόρμα. Με το που πατήσει το κουμπί καταχώρηση ενημερώνεται η βάση δεδομένων και συγκεκριμένα, προστίθεται μια καινούρια εγγραφή στον πίνακα psifodeltia με το id του συνδυασμού και το id του χρήστη , και σε περίπτωση που υπάρχουν σταυροί , αυξάνεται το πεδίο stauroi του πινάκα candidates.

3.5.2 Υλοποίηση Περιβάλλοντος Καταχωρητή

Η υλοποίηση της εφαρμογής βρίσκεται στο παράρτημα και αποτελείται από τα αρχεία

- Forma_Eisagogis_Psifodeltiou.php

  Δημιουργεί τη πρώτη φόρμα με τους συνδυασμούς

- Forma_Eisagogis_Psifodeltiou2.php

  Δημιουργεί τη δεύτερη φόρμα με τους υποψήφιους

- common.php

Περιέχει διάφορες χρήσιμες συναρτήσεις

3.6 Επισκόπηση Περιβάλλοντος Παρουσίασης Αποτελεσμάτων των Εκλογών

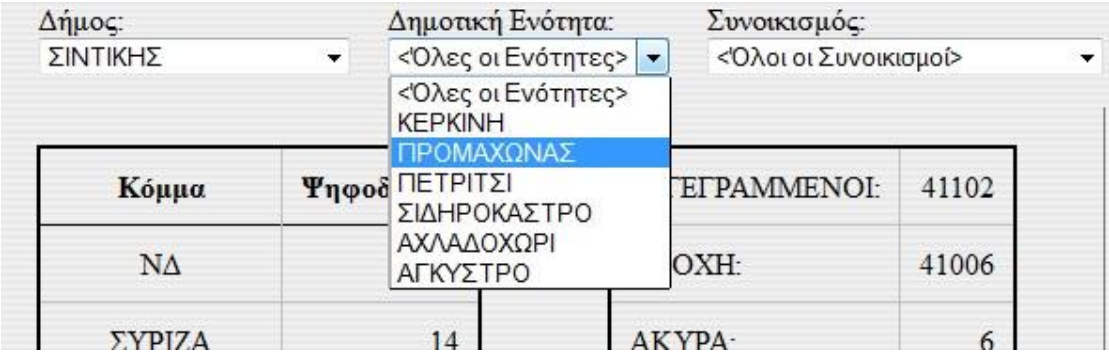
Η υλοποίηση της παρουσίασης των αποτελεσμάτων έγινε με τη βοήθεια της Php και συγκεκριμένα για τη δημιουργία γραφημάτων χρησιμοποιήθηκε η php βιβλιοθήκη GD ,η οποία είναι κατάλληλη για την δημιουργία και επεξεργασία εικόνων. Όλα τα γραφήματα και οι πίνακες υπολογίζονται δυναμικά, ανάλογα με τα στοιχεία που έχουν εισάγει οι καταχωρητές στο προηγούμενο στάδιο της εφαρμογής, δηλαδή το στάδιο καταχώρησης ψήφων στη Βάση Δεδομένων. Για να διατηρηθεί ένας σοβαρός δημόσιος χαρακτήρας αποφεύχθηκε η υπερβολική χρήση χρωμάτων

3.6.1 Παρουσίαση περιβάλλοντος

Αυτό το μέρος της εφαρμογής, σε αντίθεση με τα προηγούμενα είναι δημόσιο και δε προστατεύεται με τη χρήση κωδικών. Οποιοσδήποτε χρήστης του διαδικτύου ενδιαφέρεται μπορεί να μπει στη σελίδα που παρουσιάζονται τα συγκεντρωτικά αποτελέσματα των εκλογών.

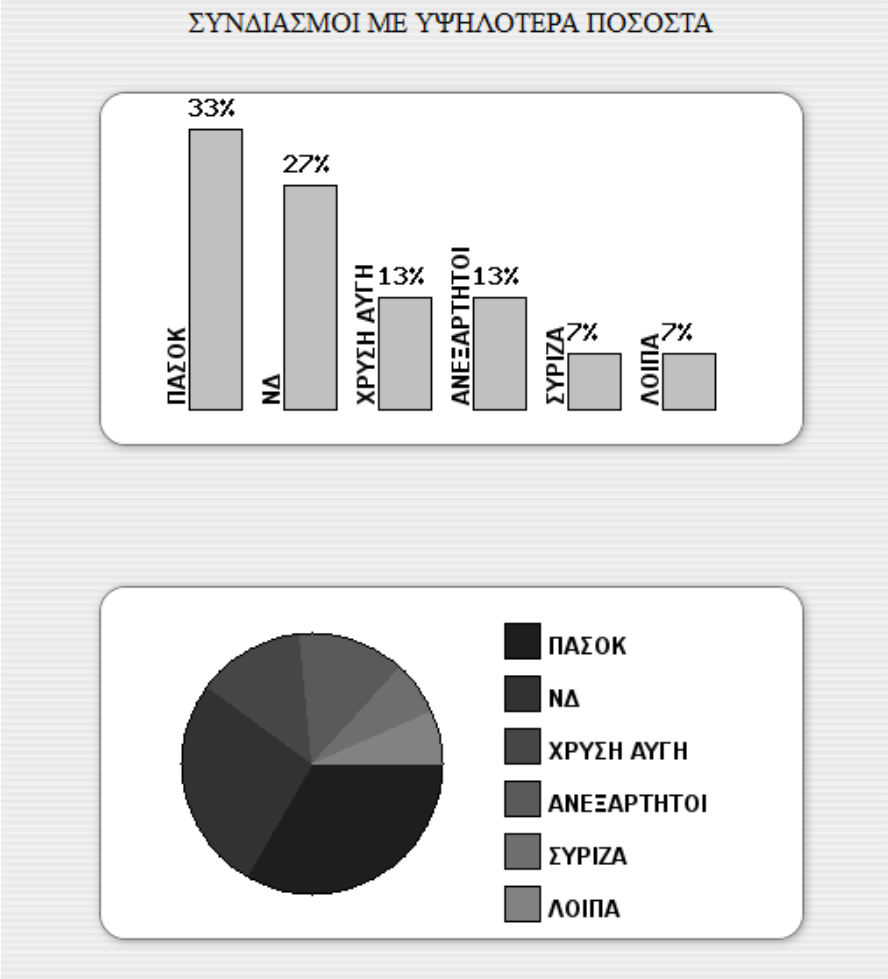
Τα αποτελέσματα βρίσκονται διαθέσιμα σε μορφή πινάκων, γραφημάτων και πίτας

Με τη χρήση τριών dropdown πεδίων ο χρήστης μπορεί να επιλέξει να δει πληροφορίες μόνο για τον δήμο, τη δημοτική ενότητα ή τον συνοικισμό που τον ενδιαφέρει. Όπως φαίνεται στην εικόνα που ακολουθεί εάν επιλεγεί κάποιος δήμος τότε στο πεδίο δημοτική ενότητα και στο πεδίο συνοικισμοί εμφανίζονται μόνο τα αντίστοιχα πεδία που ανήκουν σε αυτόν που επιλέχτηκε, περιορίζοντας έτσι την αναζήτηση του χρηστή. Αυτό είναι ιδιαίτερα χρήσιμο αν ο χρήστης ενδιαφέρεται να βρει αποτελέσματα για μια από τις 173 δημοτικές ενότητες του νομού Σερρών. Εάν δεν γίνει κάποια επιλογή τα αποτελέσματα εμφανίζοναι για ολο τον νομό Σερρών

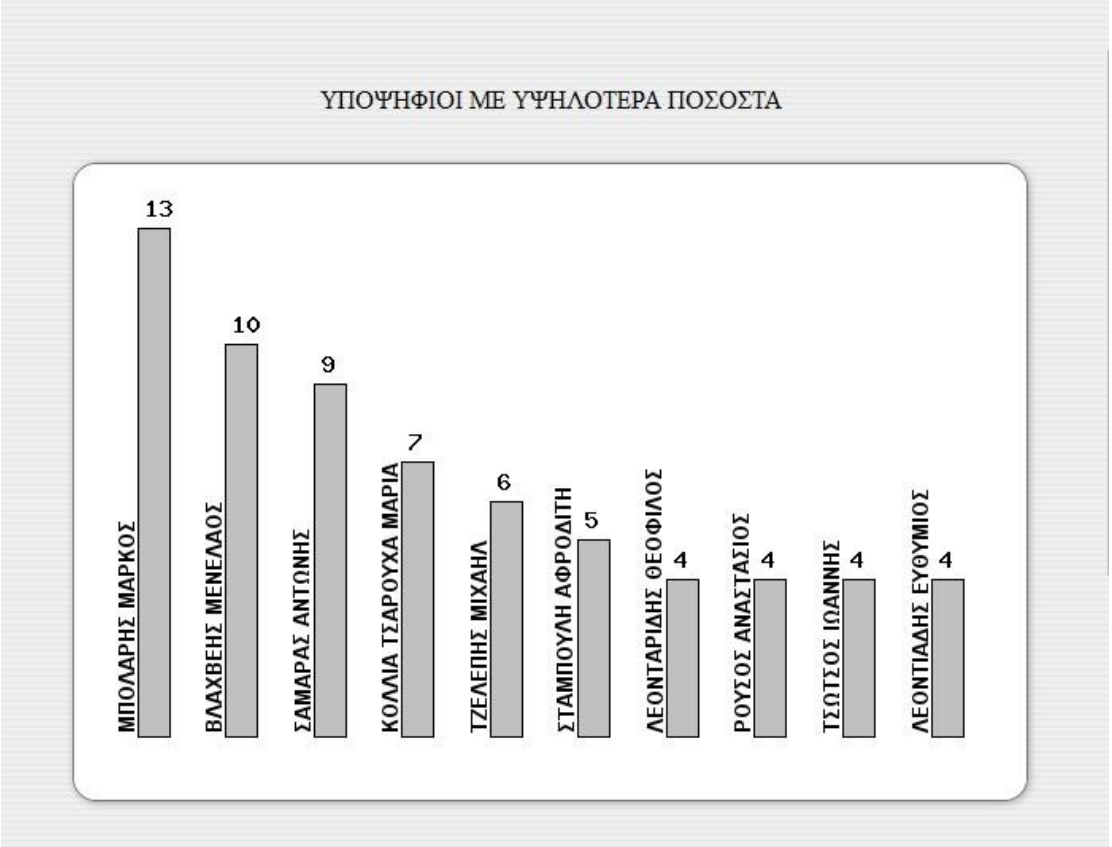


Ανάλογα με την επιλογή που έχει κάνει ο χρήστης υπολογίζονται και εμφανίζονται στοιχεία όπως οι εγγεγραμμένοι ψηφοφόροι, η αποχή, τα άκυρα και τα λευκά ψηφοδέλτια και οι πέντε (5) συνδυασμοί με τα υψηλότερα ποσοστά όπως φαίνεται στην εικόνα που ακολουθεί. Στο πεδίο λοιπά αθροίζεται το σύνολο των ψήφων των υπόλοιπων συνδυασμών.

Τα ίδια δεδομένα εμφανίζονται και με τη μορφή bar chart και pie chart



Τέλος υπολογίζονται και εμφανίζονται οι 10 υποψήφιοι που συγκέντρωσαν υψηλότερα ποσοστά ταξινομημένοι κατά φθίνοντα αριθμό ψήφων



3.6.2Υλοποίηση Περιβάλλοντος Αποτελεσμάτων

Η υλοποίηση της εφαρμογής βρίσκεται στο παράρτημα και αποτελείται από τα αρχεία

- graph.php

  Το αρχείο που δημιουργεί τις εικόνες

- ekloges.php

  το κύριο σώμα της σελίδας

- common.php

  Διάφορες συναρτήσεις για τον υπολογισμό των αποτελεσμάτων

Γενικά συμπεράσματα της Μελέτης

Σκοπός μας ήταν να δημιουργήσουμε μια εκπαιδευτικού περιεχομένου εφαρμογή η οποία θα επικοινωνεί με βάσεις δεδομένων και το περιεχόμενο της θα είναι δυναμικό σε όλα της τα επίπεδα. Δηλαδή αν και η εφαρμογή να δημιουργήθηκε για τις ανάγκες του νομού Σερρών μπορεί πολύ εύκολα, χωρίς καμία παραμετροποίηση στον κώδικα να χρησιμοποιηθεί από κάποιον άλλο νομό.

Γενικά εφάρμοσα, αλλά και ανέπτυξα περεταίρω, τις γνώσεις που είχα αποκτήσει από το ΤΕΙ πάνω στον διαδικτυακό προγραμματισμό με php και MySQL, και επιπλέον απέκτησα καινούριες γνώσεις όσον αφορά τον προγραμματισμό με JavaScript.

Η εφαρμογή αποτελεί ένα πρότυπο το οποίο εξηγεί τη βασική δομή μιας εφαρμογής γραμμένης με JavaScript Framework και τη δυναμική καταχώρηση εγγραφών στη βάση Δεδομένων. Τέλος εξηγεί πως δημιουργούνται γραφήματα με τη χρήση της PHP.

Πιθανές επεκτάσεις

- Επέκταση της δομής για όλη την επικράτεια
- Επέκταση της δομής για το ευρωπαϊκό κοινοβούλιο

ΒΙΒΛΙΟΓΡΑΦΙΑ

Για την ανάπτυξη της εφαρμογής χρησιμοποιήθηκαν πληροφορίες από:

- http://el.wikipedia.org
- http://docs.sencha.com/extjs/2.3.0/#!/api
- http://www.php.net/manual/en/ref.image.php
- Ανάπτυξη Web Εφαρμογών με PHP και MySQL Luke welling Laura Thompson εκδόσεις Μ. Γκιούρδας τέταρτη έκδοση
- Πλήρες εγχειρίδιο της HTML και CSS Laura Lemay Rafe Colburn εκδόσεις Μ. Γκιούρδας έκτη έκδοση

Παράρτημα Κώδικα

Administrator.php

```
<?php


include 'common.php';
init();
checklogin();


?>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=utf-8">
        <title> Administrator</title>
        <link rel="stylesheet" type="text/css" href="ext-
2.3.0/resources/css/ext-all.css" >
        <script type="text/javascript" src="ext-
2.3.0/adapter/ext/ext-base.js"></script>
        <script type="text/javascript" src="ext-2.3.0/ext-all-
debug.js"></script>
        <script type="text/javascript" src="admin.js"></script>
        <script type="text/javascript"
src="ux/paging/pPageSize.js"></script>
        <script type="text/javascript"
src="ux/menu/EditableItem.js"></script>
        <script type="text/javascript"
src="ux/menu/RangeMenu.js"></script>
        <script type="text/javascript"
src="ux/grid/GridFilters.js"></script>
        <script type="text/javascript"
src="ux/grid/filter/Filter.js"></script>
        <script type="text/javascript"
src="ux/grid/filter/StringFilter.js"></script>
        <script type="text/javascript"
src="ux/grid/filter/NumericFilter.js"></script>
        <link rel="stylesheet" type="text/css" href="ext-
2.3.0/examples/multiselect/MultiSelect.css"/>
        <link rel="stylesheet" type="text/css" href="ext-
2.3.0/examples/shared/examples.css" />
        <script type="text/javascript" src="ext-
2.3.0/examples/multiselect/MultiSelect.js"></script>
        <script type="text/javascript" src="ext-
2.3.0/examples/multiselect/ItemSelector.js"></script>

        <link rel="stylesheet" type="text/css" href="ext-
2.3.0/examples/shared/examples.css" />
```

```
        <style type="text/css">
            .empty .x-panel-body {
                padding-top:20px;
                text-align:center;
                font-style:italic;
                color: gray;
                font-size:11px;
            }


            body .x-panel {
                margin-bottom:20px;
            }
            .icon-grid {
                background-image:url(ext-
2.3.0/examples/shared/icons/fam/grid.png) !important;
            }
            #button-grid .x-panel-body {
                border:1px solid #99bbe8;
                border-top:0 none;
            }
            .add {
                background-image:url(ext-
2.3.0/examples/shared/icons/fam/add.gif) !important;
            }
            .option {
                background-image:url(ext-
2.3.0/examples/shared/icons/fam/plugin.gif) !important;
            }
            .remove {
                background-image:url(ext-
2.3.0/examples/shared/icons/fam/delete.gif) !important;
            }
            .refresh {
                background-image:url(ext-
2.3.0/examples/shared/icons/fam/table_refresh.png) !important;
            }
            .save {
                background-image:url(ext-
2.3.0/examples/shared/icons/save.gif) !important;
            }

        </style>
    </head>
    <body>
        <?php
        echo "UserID=".$_SESSION['Userid'].'<br>';
        ?>
        <br>

        <a href ="logout.php"> Logout</a>
        <br>
        <br>

        <div id="grid"></div>
        <div id="tab-candidate"></div>
        <div id="tab-party"></div>
        <div id="tab-users"></div>
        <div id="tab-dimoi"></div>
        <div id="tab-dimotikiEnotita"></div>
```

```
        <div id="tab-sin"></div>

    </body>
</html>
```

Admin.php

```
<?php


include 'common.php';
dbconnect();


$task = ($_POST['task']) ? ($_POST['task']) : null;
switch ($task) {

    case "readCandidate":
        showDataCandidates(); //(2)
        break;
    case "readSinikismo":
        showDataSin(); //(2)
        break;
    case "readDimotikiEnotita":
        showDataDimotikiEnotita(); //(2)
        break;
    case "readParty":
        showData('sindiasmoi'); //2
        break;
    case "readUser":
        showData('users'); //2
        break;
    case "readDimo":
        showData('dimos'); //2
        break;



    case "updateCandidate":
        saveData('candidates'); //3
        break;
    case "updateParty":
        saveData('sindiasmoi'); //3
        break;
    case "updateUser":
        saveData('users'); //3
        break;
    case "updateDimo":
        saveData('dimos'); //3
        break;
    case "updateDE":
        saveData('dimotikienotita'); //3
        break;
    case "updateSin":
        saveData('sinoikismoi'); //3
```

```
        break;


    case "deleteCandidate":
        removeData('candidates'); //4
        break;
    case "deleteParty":
        removeData('sindiasmoi'); //4
        break;
    case "deleteUser":
        removeData('users'); //4
        break;
    case "deleteDimo":
        removeData('dimos'); //4
        break;
    case "deleteDE":
        removeData('dimotikienotita'); //4
        break;
    case "deleteSinikismo":
        removeData('sinoikismoi'); //4
        break;



    case "readPartyId":
        getData('sindiasmoi'); //sub
        break;
    case "readDimosId":
        getData('dimos'); //sub
        break;
    case "readDEId":
        getData('dimotikienotita'); //sub
        break;
    default:
        echo "{failure:true}";
        break;
}

function showData($table) { //2
    $start = (integer) (isset($_POST['start']) ? $_POST['start'] :
$_GET['start']);
    $end = (integer) (isset($_POST['limit']) ? $_POST['limit'] :
$_GET['limit']);
    $sql_count = 'SELECT * FROM ' . $table;
    $sql = $sql_count . ' LIMIT ' . $start . ', ' . $end;
    $result_count = mysql_query($sql_count);
    $rows = mysql_num_rows($result_count);
    $result = mysql_query($sql);
    while ($rec = mysql_fetch_array($result, MYSQL_ASSOC)) {

        $arr[] = $rec;
    };
    if (version_compare(PHP_VERSION, "5.2", "<")) {
        require_once("./JSON.php"); //if php<5.2 need JSON class
        $json = new Services_JSON(); //instantiate new json object
        $data = $json->encode($arr);  //encode the data in json
format
    } else {
        $data = json_encode($arr);  //encode the data in json format
    }
```

```
    //$cb = isset($_GET['callback']) ? $_GET['callback'] : '';

    echo '({"total":"' . $rows . '","results":' . $data . '})';
}

//end showdata
function showDataCandidates() { //2
    $table = 'candidates';
    $start = (integer) (isset($_POST['start']) ? $_POST['start'] :
$_GET['start']);
    $end = (integer) (isset($_POST['limit']) ? $_POST['limit'] :
$_GET['limit']);
    $sql_count = 'SELECT IdCandidate, LastName, Til, sidmisi as
PartyId FROM candidates INNER JOIN sindiasmoi on
sindiasmoi.id=candidates.PartyId';
    $sql = $sql_count . ' LIMIT ' . $start . ', ' . $end;
    $result_count = mysql_query($sql_count);
    $rows = mysql_num_rows($result_count);
    $result = mysql_query($sql);
    while ($rec = mysql_fetch_array($result, MYSQL_ASSOC)) {

        $arr[] = $rec;
    };
    if (version_compare(PHP_VERSION, "5.2", "<")) {
        require_once("./JSON.php"); //if php<5.2 need JSON class
        $json = new Services_JSON(); //instantiate new json object
        $data = $json->encode($arr);  //encode the data in json
format
    } else {
        $data = json_encode($arr);  //encode the data in json format
    }

    //$cb = isset($_GET['callback']) ? $_GET['callback'] : '';

    echo '({"total":"' . $rows . '","results":' . $data . '})';
}

function showDataDimotikiEnotita() { //2
    $start = (integer) (isset($_POST['start']) ? $_POST['start'] :
$_GET['start']);
    $end = (integer) (isset($_POST['limit']) ? $_POST['limit'] :
$_GET['limit']);
    $sql_count = 'SELECT IdDE,DimotikiEnotita,Dimos as IdDimos FROM
dimotikienotita INNER JOIN dimos on
dimos.IdDimos=dimotikienotita.IdDimos';
    $sql = $sql_count . ' LIMIT ' . $start . ', ' . $end;
    $result_count = mysql_query($sql_count);
    $rows = mysql_num_rows($result_count);
    $result = mysql_query($sql);
    while ($rec = mysql_fetch_array($result, MYSQL_ASSOC)) {

        $arr[] = $rec;
    };
    if (version_compare(PHP_VERSION, "5.2", "<")) {
        require_once("./JSON.php"); //if php<5.2 need JSON class
        $json = new Services_JSON(); //instantiate new json object
        $data = $json->encode($arr);  //encode the data in json
format
    } else {
        $data = json_encode($arr);  //encode the data in json format
```

```
    }

    //$cb = isset($_GET['callback']) ? $_GET['callback'] : '';

    echo '({"total":"' . $rows . '","results":' . $data . '})';
}

function showDataSin() { //2
    $start = (integer) (isset($_POST['start']) ? $_POST['start'] :
$_GET['start']);
    $end = (integer) (isset($_POST['limit']) ? $_POST['limit'] :
$_GET['limit']);
    $sql_count = 'SELECT IdSin, Sinikismoi,ArithmosPsifoforon,
DimotikiEnotita as IdDE FROM sinoikismoi INNER JOIN dimotikienotita
on dimotikienotita.IdDE=sinoikismoi.IdDE';
    $sql = $sql_count . ' LIMIT ' . $start . ', ' . $end;
    $result_count = mysql_query($sql_count);
    $rows = mysql_num_rows($result_count);
    $result = mysql_query($sql);
    while ($rec = mysql_fetch_array($result, MYSQL_ASSOC)) {

        $arr[] = $rec;
    };
    if (version_compare(PHP_VERSION, "5.2", "<")) {
        require_once("./JSON.php"); //if php<5.2 need JSON class
        $json = new Services_JSON(); //instantiate new json object
        $data = $json->encode($arr);  //encode the data in json
format
    } else {
        $data = json_encode($arr);  //encode the data in json format
    }

    //$cb = isset($_GET['callback']) ? $_GET['callback'] : '';

    echo '({"total":"' . $rows . '","results":' . $data . '})';
}

function getData($table) {
    $sql = 'SELECT * FROM ' . $table;
    $result = mysql_query($sql);

    while ($rec = mysql_fetch_array($result, MYSQL_ASSOC)) {
        $arr[] = $rec;
    };

    if (version_compare(PHP_VERSION, "5.2", "<")) {
        require_once("./JSON.php"); //if php<5.2 need JSON class
        $json = new Services_JSON(); //instantiate new json object
        $data = $json->encode($arr);  //encode the data in json
format
    } else {
        $data = json_encode($arr);  //encode the data in json format
    }

    $cb = isset($_GET['callback']) ? $_GET['callback'] : '';

    echo $cb . '({"results":' . $data . '})';
}

//end getData
```

```
function saveData($table) { //3
    $key = $_POST['key'];
    $id = (integer) mysql_real_escape_string($_POST['keyID']);
    $field = $_POST['field'];
    $value = $_POST['value'];
    $newRecord = $id == 0 ? 'yes' : 'no';

    if ($newRecord == 'yes') {
        $query = 'INSERT INTO `' . $table . '` (`' . $field . '`)
VALUES (\'' . $value . '\')';
    } else {
        $query = 'UPDATE `' . $table . '` SET `' . $field . '` = \''
. $value . '\' WHERE `' . $key . '` = ' . $id;
    }

    //save data to database
    $result = mysql_query($query);
    $rows = mysql_affected_rows();

    if ($rows > 0) {
        if ($newRecord == 'yes') {
            $newID = mysql_insert_id();
            echo "{success:true, newID:$newID}";
        } else {
            echo "{success:true}";
        }
    } else {
        echo "{success:false}";
    }
}

//end save data

function removeData($table) {

    $key = $_POST['key'];
    $arr = $_POST['id'];
    $count = 0;


    if (version_compare(PHP_VERSION, "5.2", "<")) {
        require_once("./JSON.php");
        $json = new Services_JSON();
        $selectedRows = $json->decode(stripslashes($arr));
    } else {
        $selectedRows = json_decode(stripslashes($arr));
    }

    foreach ($selectedRows as $row_id) {
        $id = (integer) $row_id;
        $query = 'DELETE FROM `' . $table . '` WHERE `' . $key . '` =
' . $id;
        $result = mysql_query($query); //returns number of rows
deleted
        if ($result)
            $count++;
    }

    if ($count) { //only checks if the last record was deleted,
others may have failed
        $cb = isset($_GET['callback']) ? $_GET['callback'] : '';
```

```
        $response = array('success' => $count, 'del_count' =>
$count);


        if (version_compare(PHP_VERSION, "5.2", "<")) {
            $json_response = $json->encode($response);
        } else {
            $json_response = json_encode($response);
        }

        echo $cb . $json_response;
//        echo '{success: true, del_count: '.$count.'}';
    } else {
        echo '{failure: true}';
    }
}

//end removeData
?>
```

Admin.Js

```
Ext.BLANK_IMAGE_URL = 'ext-2.3.0/resources/images/default/s.gif';
Ext.namespace('myNameSpace');


var colModelP; //καθορισμός των στηλών για τους Συνδυασμούς
var colModelC; //καθορισμός των στηλών για τους Υποψήφιους
var colModelU;//καθορισμός των στηλών για τους Χρήστες
var colModelDimos;//καθορισμός των στηλών για τους Δήμους
var colModelDE;//καθορισμός των στηλών για τις Δημοτικές Ενότητες
var colModelS;//καθορισμός των στηλών για τους Συνοικισμούς

var SindiasmoiRecordObj; //Record Object για τους Συνδυασμούς
var CandidatesRecordObj;//Record Object για τους Υποψήφιους
var UsersRecordObj;//Record Object για τους Χρήστες
var DimoiRecordObj;//Record Object για τους Δήμους
var DERecordObj;//Record Object για τις Δημοτικές Ενότητες
var SinikismoiRecordObj;//Record Object για τους Συνοικισμούς

var dsSindiasmoi; //Data Store για τους Συνδυασμούς
var dsCandidates;//Data Store για τους Υποψήφιους
var dsPartyId;//Δευτερεύον Data Store για τους Υποψήφιους
var dsUsers;//Data Store για τους Χρήστες
var dsDimoi;//Data Store για τους Δήμους
var dsDE//Data Store για τις Δημοτικες Ενότητες
var dsDimosId;//Δευτερεύον Data Store για τις Δημοτικές Ενότητες
var dsSin;//Data Store για τους Συνοικισμούς
var dsDEId;//Δευτερεύον Data Store για τους Συνοικισμούς


var gridPartys; //Πίνακας(grid) για τους Συνδυασμούς
var gridCandidates;//Πίνακας(grid) για τους Υποψήφιους
var gridUsers;//Πίνακας(grid) για τους Χρήστες
var gridDimoi;//Πίνακας(grid) για τους Δήμους
var gridDE;//Πίνακας(grid) για τις Δημοτικές Ενότητες
var gridSin;//Πίνακας(grid) για τους Συνοικισμούς
```

```
var readerParty; //Reader για τους Συνδιασμούς
var readerCandidate;//Reader για τους Υποψήφιους
var readerUsers;//Reader για τους Χρήστες
var readerDimos;//Reader για τους Δήμους
var readerDE;//Reader για τις Δημοτικές Ενότητες
var readerSin;//Reader για τους Συνοικισμούς

var filters; //φίλτρα
var pagingPlugin;

////////////////////////////////////////////////////////////////////////
////////


var setupDataSourceP = function() { //1.0

   //1.1
    SindiasmoiRecordObj = Ext.data.Record.create([
        {
            name: 'id'
        },
        {
            name: 'onoma',
            sortDir: 'ASC',
            sortType: 'asUCString'
        },
        {
            name: 'tilefono',
            sortDir: 'ASC',
            sortType: 'asUCString'
        },
        {
            name: 'sidmisi',
            sortDir: 'ASC',
            sortType: 'asUCString'
        }
    ]);



    //DEFINE READER 1.2
    readerParty = new Ext.data.JsonReader(
            {
                root: 'results',
                totalProperty: 'total',
                id: 'id'
            },
    SindiasmoiRecordObj
            );//end readerParty


    //CREATE DATASTORE 1.3
    dsSindiasmoi = new Ext.data.Store({
        proxy: new Ext.data.HttpProxy({
            url: 'admin.php',
            method: 'POST'
        }),
        baseParams: {
            task: "readParty"
        },
```

```
        reader: readerParty,
        sortInfo: {
            field: 'id',
            direction: "ASC"
        }
    });//end dsSindiasmoi


};//end setupDataSourceP

var setupDataSourceC = function() {
    CandidatesRecordObj = Ext.data.Record.create([
        {
            name: 'IdCandidate'
        },
        {
            name: 'LastName',
            sortDir: 'ASC',
            sortType: 'asUCString'
        },
        {
            name: 'Til',
            sortDir: 'ASC',
            sortType: 'asUCString'

        },
        {
            name: 'PartyId',
            sortDir: 'ASC',
            sortType: 'asUCString'

        }
    ]);

    readerCandidate = new Ext.data.JsonReader
            {
                root: 'results',
                totalProperty: 'total',
                id: 'IdCandidate'
            },
    CandidatesRecordObj
            ); //end readerCandidate

    dsCandidates = new Ext.data.Store({
        proxy: new Ext.data.HttpProxy({
            url: 'admin.php',
            method: 'POST'
        }),
        baseParams: {
            task: "readCandidate"
        },
        reader: readerCandidate,
        sortInfo: {
            field: 'IdCandidate',
            direction: "ASC"
        }
    });//end dsCandidates

    dsPartyId = new Ext.data.Store({
        proxy: new Ext.data.HttpProxy({
            url: 'admin.php',
```

```
            method: 'POST'
        }),
        baseParams: {
            task: "readPartyId"
        },
        reader: new Ext.data.JsonReader({
            root: 'results',
            id: 'PartyId'
        }, [
            {name: 'id'},
            {name: 'sidmisi'}]
                ),
        sortInfo: {
            field: 'sidmisi',
            direction: "ASC"
        }
    });//end dsPartyId
};//End SetupDatasourceC

var setupDataSourceU = function() { //1.0
    UsersRecordObj = Ext.data.Record.create([
        {
            name: 'UserId'
        },
        {
            name: 'Username',
            sortDir: 'ASC',
            sortType: 'asUCString'
        },
        {
            name: 'Epitheto',
            sortDir: 'ASC',
            sortType: 'asUCString'
        },
        {
            name: 'IdS',
            sortDir: 'ASC',
            sortType: 'asUCString'
        },
        {
            name: 'Password',
            sortDir: 'ASC',
            sortType: 'asUCString'
        },
        {
            name: 'OmadaErgasias',
            sortDir: 'ASC',
            sortType: 'asUCString'
        }
    ]);



    //DEFINE READER 1.2
    readerUsers = new Ext.data.JsonReader(
            {
                root: 'results',
                totalProperty: 'total',
                id: 'UserId'
            },
    UsersRecordObj
```

```
            );//end readerUsers


    //CREATE DATASTORE 1.3
    dsUsers = new Ext.data.Store({
        proxy: new Ext.data.HttpProxy({
            url: 'admin.php',
            method: 'POST'
        }),
        baseParams: {
            task: "readUser"
        },
        reader: readerUsers,
        sortInfo: {
            field: 'UserId',
            direction: "ASC"
        }
    });//end dsUsers


};//end setupDataSourceU

var setupDataSourceDimoi = function() { //1.0
    DimoiRecordObj = Ext.data.Record.create([
        {
            name: 'IdDimos'
        },
        {
            name: 'Dimos',
            sortDir: 'ASC',
            sortType: 'asUCString'
        }

    ]);



    //DEFINE READER 1.2
    readerDimos = new Ext.data.JsonReader(
            {
                root: 'results',
                totalProperty: 'total',
                id: 'IdDimos'
            },
    DimoiRecordObj
            );//end readerDimwn


    //CREATE DATASTORE 1.3
    dsDimoi = new Ext.data.Store({
        proxy: new Ext.data.HttpProxy({
            url: 'admin.php',
            method: 'POST'
        }),
        baseParams: {
            task: "readDimo"
        },
        reader: readerDimos,
        sortInfo: {
            field: 'IdDimos',
            direction: "ASC"
```

```
        }
    });//end dsDimoi


};//end DataSourceDimoi

var DataSourceDimotikiEnotita = function() {
    DERecordObj = Ext.data.Record.create([
        {
            name: 'IdDE'
        },
        {
            name: 'DimotikiEnotita',
            sortDir: 'ASC',
            sortType: 'asUCString'
        },
        {
            name: 'IdDimos',
            sortDir: 'ASC',
            sortType: 'asUCString'

        }
    ]);

    readerDE = new Ext.data.JsonReader(
            {
                root: 'results',
                totalProperty: 'total',
                id: 'IdDE'
            },
    DERecordObj
            ); //end readerDE

    dsDE = new Ext.data.Store({
        proxy: new Ext.data.HttpProxy({
            url: 'admin.php',
            method: 'POST'
        }),
        baseParams: {
            task: "readDimotikiEnotita"
        },
        reader: readerDE,
        sortInfo: {
            field: 'IdDE',
            direction: "ASC"
        }
    });//end dsDE

    dsDimosId = new Ext.data.Store({
        proxy: new Ext.data.HttpProxy({
            url: 'admin.php',
            method: 'POST'
        }),
        baseParams: {
            task: "readDimosId"
        },
        reader: new Ext.data.JsonReader({
            root: 'results',
            id: 'IdDimos'//dimotikienotita
        }, [
            {name: 'IdDimos'}, //dimos
```

```
            {name: 'Dimos'}]
                ),
        sortInfo: {
            field: 'Dimos',
            direction: "ASC"
        }
    });//end dsPDimosId
};

var DataSourceSinikismoi = function() {
    SinikismoiRecordObj = Ext.data.Record.create([
        {
            name: 'IdSin'
        },
        {
            name: 'Sinikismoi',
            sortDir: 'ASC',
            sortType: 'asUCString'
        },
        {
            name: 'IdDE',
            sortDir: 'ASC',
            sortType: 'asUCString'

        }, {
            name: 'ArithmosPsifoforon',
            sortDir: 'ASC',
            sortType: 'asUCString'
        }
    ]);

    readerSin = new Ext.data.JsonReader(
            {
                root: 'results',
                totalProperty: 'total',
                id: 'IdSin'
            },
    SinikismoiRecordObj
            ); //end readerSin

    dsSin = new Ext.data.Store({
        proxy: new Ext.data.HttpProxy({
            url: 'admin.php',
            method: 'POST'
        }),
        baseParams: {
            task: "readSinikismo"
        },
        reader: readerSin,
        sortInfo: {
            field: 'Sinikismoi',
            direction: "ASC"
        }
    });//end dsSin

    dsDEId = new Ext.data.Store({
        proxy: new Ext.data.HttpProxy({
            url: 'admin.php',
            method: 'POST'
        }),
        baseParams: {
```

```
            task: "readDEId"
        },
        reader: new Ext.data.JsonReader({
            root: 'results',
            id: 'IdDE'
        }, [
            {name: 'IdDE'},
            {name: 'DimotikiEnotita'}]
                ),
        sortInfo: {
            field: 'DimotikiEnotita',
            direction: "ASC"
        }
    });//end dsDEId
};

/////////////////////////////////////////////////
/////////////////////END OF DATASOURCE///////////
/////////////////////////////////////////////////

var getColumnModelP = function() {
    if (!colModelP) {
        colModelP = new Ext.grid.ColumnModel([
            {
                dataIndex: 'id',
                header: "ID",
                sortable: true
            }, {
                dataIndex: 'onoma',
                header: "Ονομα",
                width: 600,
                sortable: true,
                editor: new Ext.form.TextField({
                    allowBlank: false
                })
            }, {
                dataIndex: 'sidmisi',
                header: "Συντόμευση",
                width: 300,
                sortable: true,
                editor: new Ext.form.TextField({
                    allowBlank: false
                })
            }, {
                dataIndex: 'tilefono',
                header: "Τηλέφωνο",
                width: 200,
                sortable: true,
                editor: new Ext.form.NumberField({
                    allowBlank: false,
                    maxLength: '14',//έλεγχος απο 10 εως 12 ψηφία
                    minLength: '10'
                })
            }
        ]);//end colModelP
        this.filters = new Ext.ux.grid.GridFilters({
            local: true,
            filters: [
                {
                    dataIndex: 'tilefono',// έλεγχος οτι στο πεδίο
τηλέφωνο θα εχει μόνο νούμερα
```

```
                    type: 'numeric'
                }
            ]
        });
    }//end if colModelP
    return colModelP;
}//end GetColumnModelP

var getColumnModelC = function() {
    if (!colModelC) {

        dsPartyId.load();
        colModelC = new Ext.grid.ColumnModel([
            {
                dataIndex: 'IdCandidate',
                header: "ID",
                width: 100,
                sortable: true
            }, {
                dataIndex: 'LastName',
                header: 'Ονοματεπώνυμο',
                width: 650,
                sortable: true,
                editor: new Ext.form.TextField({
                    allowBlank: false
                })
            }, {
                dataIndex: 'Til',
                header: 'Τηλέφωνο',
                width: 250,
                sortable: true,
                editor: new Ext.form.NumberField({
                    allowBlank: false,
                    maxLength: '12',
                    minLength: '10'
                })
            }, {
                dataIndex: 'PartyId',
                header: 'Κόμμα',
                width: 260,
                sortable: true,
                editor: new Ext.form.ComboBox({
                    typeAhead: false,
                    triggerAction: 'all',
                    lazyRender: true,
                    store: dsPartyId,
                    displayField: 'sidmisi',
                    valueField: 'id'

                })
            }
        ]); //end colmodelc

        this.filters = new Ext.ux.grid.GridFilters({
            local: true,
            filters: [
                {
                    dataIndex: 'Til',
                    type: 'numeric'
                }
            ]
```

```
        });

    }//endif
    return colModelC;
}//end getColumnModelC

var getColumnModelU = function() {
    if (!colModelU) {//Create Column Model
        colModelU = new Ext.grid.ColumnModel([
            {
                dataIndex: 'UserId',
                header: "ID",
                sortable: true
            }, {
                dataIndex: 'Username',
                header: "Username",
                width: 200,
                sortable: true,
                editor: new Ext.form.TextField({
                    allowBlank: false
                })
            }, {
                dataIndex: 'Epitheto',
                header: "Ονοματεπώνυμο",
                width: 200,
                sortable: true,
                editor: new Ext.form.TextField({
                    allowBlank: false
                })
            }, {
                dataIndex: 'IdS',
                header: "Συνοικισμός",
                width: 400,
                sortable: true,
                editor: new Ext.form.TextField({
                    allowBlank: false
                })
            }, {
                dataIndex: 'Password',
                header: "Κωδικός",
                width: 200,
                sortable: true,
                editor: new Ext.form.TextField({
                    allowBlank: false

                })
            }, {
                dataIndex: 'OmadaErgasias',
                header: "is admin",
                width: 150,
                sortable: true,
                editor: new Ext.form.NumberField({
                    allowBlank: false,
                    maxValue: '1',
                    minValue: '0'
                })
            }
        ]);//end colModelU

        this.filters = new Ext.ux.grid.GridFilters({
            local: true,
```

```
            filters: [
                {
                    dataIndex: 'OmadaErgasias',
                    type: 'numeric'
                }
            ]
        });
    }//end if colModelU
    return colModelU;
}//end GetColumnModelU

var getColumnModelDimos = function() {
    if (!colModelDimos) {//Create Column Model
        colModelDimos = new Ext.grid.ColumnModel([
            {
                dataIndex: 'IdDimos',
                header: "ID",
                width: 200,

                sortable: true
            }, {
                dataIndex: 'Dimos',
                header: "Δήμος",
                width: 800,
                sortable: true,
                editor: new Ext.form.TextField({
                    allowBlank: false
                })
            }
        ]);//end colModelDimos
    }//end if colModelDimos
    return colModelDimos;
}//end GetColumnModelDimos

var ColumnModelDE = function() {
    if (!colModelDE) {

        dsDimosId.load();
        colModelDE = new Ext.grid.ColumnModel([
            {
                dataIndex: 'IdDE',
                header: "ID",
                sortable: true
            }, {
                dataIndex: 'DimotikiEnotita',
                header: 'Δημοτική ενότητα',
                width: 600,
                sortable: true,
                editor: new Ext.form.TextField({
                    allowBlank: false
                })
            }, {
                dataIndex: 'IdDimos',
                header: 'Δήμος',
                width: 400,
                sortable: true,
                editor: new Ext.form.ComboBox({
                    typeAhead: false,
                    triggerAction: 'all',
                    lazyRender: true,
                    store: dsDimosId,
```

```
                    displayField: 'Dimos',
                    valueField: 'IdDimos'

                })
            }
        ]); //end colmodelc


    }//endif
    return colModelDE;
}

var ColumnModelS = function() {
    if (!colModelS) {

        dsDEId.load();
        colModelS = new Ext.grid.ColumnModel([
            {
                dataIndex: 'IdSin',
                header: "ID",
                sortable: true
            }, {
                dataIndex: 'Sinikismoi',
                header: 'Συνοικισμοί',
                width: 500,
                sortable: true,
                editor: new Ext.form.TextField({
                    allowBlank: false
                })
            }, {
                dataIndex: 'IdDE',
                header: 'Δημοτική Ενότητα',
                width: 400,
                sortable: true,
                editor: new Ext.form.ComboBox({
                    typeAhead: false,
                    triggerAction: 'all',
                    lazyRender: true,
                    store: dsDEId,
                    displayField: 'DimotikiEnotita',
                    valueField: 'IdDE'

                })
            }, {
                dataIndex: 'ArithmosPsifoforon',
                header: 'Εγγεγραμένοι Ψηφοφόροι',
                width: 200,
                sortable: true,
                editor: new Ext.form.NumberField({
                    allowBlank: false
                })
            }
        ]); //end colmodelS

        this.pagingPlugin = new Ext.ux.Andrie.pPageSize({
            afterText: 'records at a time'
        });
    }//endif
    return colModelS;
}
```

```
////////////////////////////////////////////////////////
//////////////////////END OF COLUMNMODEL////////////////
////////////////////////////////////////////////////////

var buildGridP = function() {

    function addRecordP() {
        var r = new SindiasmoiRecordObj({
            id: 0,
            onoma: '',
            sidmisi: '',
            tilefono: ''

        });

        gridPartys.stopEditing();
        dsSindiasmoi.insert(0, r);
        gridPartys.startEditing(0, 1);
    } // end addRecordP

    //function for deleting records
    function deleteRecord(btn) {
        if (btn == 'yes')
        {
            var selectedRows = gridPartys.selModel.selections.items;
            var selectedKeys = gridPartys.selModel.selections.keys;
            var encoded_keys = Ext.encode(selectedKeys);
Ext.Ajax.request(
                    {
                        waitMsg: 'Saving changes...',
                        url: 'admin.php',
                        params: {
                            task: "deleteParty",
                            id: encoded_keys,
                            key: 'id'
                        },
                        failure: function(response, options) {
                            Ext.MessageBox.alert('Warning',
'Oops...');
                            dsSindiasmoi.rejectChanges();
},
                        success: function(response, options) {
                            dsSindiasmoi.reload();
                        }
                    } //end Ajax request config
            )// end Ajax request initialization
        }//end if click 'yes' on button
    } // end deleteRecord1
    //handler for deleting records
    function handleDelete() {

        //returns array of selected rows ids only
        var selectedKeys = gridPartys.selModel.selections.keys;
        if (selectedKeys.length > 0)
        {
            Ext.MessageBox.confirm('Message', 'Do you really want to
delete selection?', deleteRecord);
        }
        else
        {
```

```
            Ext.MessageBox.alert('Message', 'Please select at least
one item to delete');
        }//end if/else block
    } // end handleDelete

    function handleEdit(editEvent) {
        var gridField = editEvent.field;
        updateDB(editEvent);
    }

    function refreshGrid() {
        dsSindiasmoi.reload();
    }

    function updateDB(oGrid_Event) {

        if (oGrid_Event.value instanceof Date) {
            var fieldValue = oGrid_Event.value.format('Y-m-d H:i:s');
        } else
        {
            var fieldValue = oGrid_Event.value;
        }

        //submit to server
        Ext.Ajax.request(
                {
                    waitMsg: 'Saving changes...',
                    url: 'admin.php',
                    params: {
                        task: "updateParty",
                        key: 'id',
                        keyID: oGrid_Event.record.data.id,
                        field: oGrid_Event.field,
                        value: fieldValue,
                        originalValue: oGrid_Event.record.modified

                    }, //end params

                    failure: function(response, options) {
                        Ext.MessageBox.alert('Warning', 'Oops...');
                      }, //end failure block

                    success: function(response, options) {
                      if (oGrid_Event.record.data.id == 0) {
                      var responseData =
      Ext.util.JSON.decode(response.responseText);
                            var newID = responseData.newID;
oGrid_Event.record.set('newRecord', 'no');

                            //Assign the id to the record
                            oGrid_Event.record.set('id', newID);
                            dsSindiasmoi.commitChanges();

                            //var whatIsTheID =
oGrid_Event.record.modified;

                            //not a new record so just commit changes
                        } else {
                            //commit changes (removes the red
triangle
                            //which indicates a 'dirty' field)
```

```
                            dsSindiasmoi.commitChanges();
                        }
                    }//end success block
                }//end request config
        ); //end request
    } //end updateDB

    //CREATE THE GRID 3.2
    gridPartys = new Ext.grid.EditorGridPanel({
        clicksToEdit: 2,
        colModel: getColumnModelP(),
        verticalScrollerType: 'paginggridscroller',
        invalidateScrollerOnRefresh: false,
        disableSelection: true,
        frame: true,
        // autoExpandColumn:'onoma',
        height: 650,
        width: 1000,
        id: 'SindiasmoiGrid',
        plugins: filters,
        loadMask: true,
        selModel: new Ext.grid.RowSelectionModel({
            singleSelect: false
        }),
        store: dsSindiasmoi,
        title: 'Λίστα Συνδιασμών',
        trackMouseOver: true,
        bbar: new Ext.PagingToolbar({
            pageSize: 20,
            store: dsSindiasmoi,
            displayInfo: true,
            displayMsg: 'Displaying topics {0} - {1} of {2}',
            emptyMsg: "No data to display",
            items: []
        }), //end bbar
        tbar: [//Add a top bar
            {
                text: 'Add Record',
                tooltip: 'Click to Add a row',
                iconCls: 'add',
                handler: addRecordP
            }, '-', //add a separator
            {
                text: 'Delete Selected',
                tooltip: 'Click to Delete selected row(s)',
                handler: handleDelete,
                iconCls: 'remove'
            }, '->', // next fields will be aligned to the right
            {
                text: 'Refresh',
                tooltip: 'Click to Refresh the table',
                handler: refreshGrid,
                iconCls: 'refresh'
            }
        ]
    });//end grid ,EditorGridPanel

    //3.3  listeners
    gridPartys.addListener('afteredit', handleEdit)
}//END FUNCTION BULID GRID Partys
```

```
var buildGridC = function() {

    function addRecordC() {
        var nr = new CandidatesRecordObj({
            IdCandidate: 0,
            LastName: '',
            Til: '',
            PartyId: ''

        });

        gridCandidates.stopEditing();//stops any acitve editing fix
        dsCandidates.insert(0, nr); //1st arg is index,2nd arg is
Ext.data.Record[] records
        gridCandidates.startEditing(0, 1); //fix
    } // end addRecordC

    //function for deleting records
    function deleteRecordC(btn) {
        if (btn == 'yes')
        {
            var selectedRows =
gridCandidates.selModel.selections.items;
            var selectedKeys =
gridCandidates.selModel.selections.keys; //returns sel. row's id
            var encoded_keys = Ext.encode(selectedKeys);//encode
array into json
            Ext.Ajax.request(
                    {
                        waitMsg: 'Saving changes...',
                        url: 'admin.php',
                        params: {
                            task: "deleteCandidate",
                            id: encoded_keys,
                            key: 'IdCandidate'   //ok
                        },
                        failure: function(response, options) {
                            Ext.MessageBox.alert('Warning',
'Oops...');
                            dsCandidates.rejectChanges();//undo any
changes
                        },
                        success: function(response, options) {
                            dsCandidates.reload();
                        }
                    } //end Ajax request config
            )// end Ajax request initialization
        }//end if click 'yes' on button
    } // end deleteRecordC
    //handler for deleting records
    function handleDeleteC() {

        //returns array of selected rows ids only
        var selectedKeys = gridCandidates.selModel.selections.keys;
        if (selectedKeys.length > 0)
        {
            Ext.MessageBox.confirm('Message', 'Do you really want to
delete selection?', deleteRecordC);
        }
        else
        {
```

```
            Ext.MessageBox.alert('Message', 'Please select at least
one item to delete');
        }//end if/else block
    } // end handleDeletec

    function handleEditC(editEvent) {
        var gridField = editEvent.field;
        updateDBC(editEvent);
    }

    function refreshGridC() {
        dsCandidates.reload();
    }

    function updateDBC(oGrid_Event) {

        if (oGrid_Event.value instanceof Date) {
            var fieldValue = oGrid_Event.value.format('Y-m-d H:i:s');
        } else
        {
            var fieldValue = oGrid_Event.value;
        }

        //submit to server
        Ext.Ajax.request(
                {
                    waitMsg: 'Saving changes...',
                    url: 'admin.php',
                    params: {
                        task: "updateCandidate",
                        key: 'IdCandidate', //ok
                        keyID: oGrid_Event.record.data.IdCandidate,
                        field: oGrid_Event.field, //the column name
                        value: fieldValue, //the updated value
                        originalValue: oGrid_Event.record.modified

                    }, //end params

                    failure: function(response, options) {
                        Ext.MessageBox.alert('Warning', 'Oops...');
                     }, //end failure block

                    success: function(response, options) {
                        //if this is a new record need special
handling
                        if (oGrid_Event.record.data.IdCandidate == 0)
{
                            var responseData =
Ext.util.JSON.decode(response.responseText);//passed back from server

                            //Extract the ID provided by the server
                            var newID = responseData.newID;
                            //oGrid_Event.record.id = newID;

                            //Reset the indicator since update
succeeded
                            oGrid_Event.record.set('newRecord',
'no');

                            //Assign the id to the record
```

```
                                oGrid_Event.record.set('IdCandidate',
newID);
                                dsCandidates.commitChanges();

                            } else {
                                dsCandidates.commitChanges();
                            }
                        }//end success block
                    }//end request config
            ); //end request
        } //end updateDBC

        //CREATE THE GRID 3.2
        gridCandidates = new Ext.grid.EditorGridPanel({
            clicksToEdit: 2,
            store: dsCandidates,
            title: 'Λίστα Υποψηφίων',
            colModel: getColumnModelC(),
            verticalScrollerType: 'paginggridscroller',
            invalidateScrollerOnRefresh: false,
            disableSelection: true,
            frame: true,
            height: 650,
            id: 'CandidatesGrid',
            plugins: filters,
            loadMask: true, //use true to mask the grid while loading
(default = false)
            selModel: new Ext.grid.RowSelectionModel({
                singleSelect: false
            }), //true to limit row selection to 1 row})

            trackMouseOver: true,
            width: 1000,
            //Add a bottom bar
            bbar: new Ext.PagingToolbar({
                pageSize: 50,
                store: dsCandidates,
                displayInfo: true,
                displayMsg: 'Displaying topics {0} - {1} of {2}',
                emptyMsg: "No data to display", //display message when no
records found
                items: []
            }), //end bbar
            tbar: [//Add a top bar
                {
                    text: 'Add Record',
                    tooltip: 'Click to Add a row',
                    iconCls: 'add',
                    handler: addRecordC
                }, '-', //add a separator
                {
                    text: 'Delete Selected',
                    tooltip: 'Click to Delete selected row(s)',
                    handler: handleDeleteC,
                    iconCls: 'remove'
                }, '->', // next fields will be aligned to the right
                {
                    text: 'Refresh',
                    tooltip: 'Click to Refresh the table',
                    handler: refreshGridC,
                    iconCls: 'refresh'
```

```
            }
        ]
    });//end grid ,EditorGridPanel

    //3.3 add listeners
    gridCandidates.addListener('afteredit', handleEditC);
}//END FUNCTION BULID GRIDc

var buildGridU = function() {

    function addRecordU() {
        var u = new UsersRecordObj({
            UserId: 0,
            Username: '',
            Epitheto: '',
            IdPE: '',
            Password: '',
            OmadaErgasias: ''

        });

        gridUsers.stopEditing();//stops any acitve editing fix
        dsUsers.insert(0, u); //1st arg is index,2nd arg is
Ext.data.Record[] records
        gridUsers.startEditing(0, 1); //fix
    } // end addRecordU

    //function for deleting records
    function deleteRecordU(btn) {
        if (btn == 'yes')
        {
            var selectedRows = gridUsers.selModel.selections.items;
            var selectedKeys = gridUsers.selModel.selections.keys;
//returns sel. row's id
            var encoded_keys = Ext.encode(selectedKeys);//encode
array into json
            Ext.Ajax.request(
                    {
                        waitMsg: 'Saving changes...',
                        url: 'admin.php',
                        params: {
                            task: "deleteUser",
                            id: encoded_keys,
                            key: 'UserId'
                        },
                        failure: function(response, options) {
                            Ext.MessageBox.alert('Warning',
'Oops...');
                            dsUsers.rejectChanges();//undo any
changes
                        },
                        success: function(response, options) {
                            dsUsers.reload();
                        }
                    } //end Ajax request config
            )// end Ajax request initialization
        }//end if click 'yes' on button
    } // end deleteRecord1
    //handler for deleting records
    function handleDeleteU() {
```

```
        //returns array of selected rows ids only
        var selectedKeys = gridUsers.selModel.selections.keys;
        if (selectedKeys.length > 0)
        {
            Ext.MessageBox.confirm('Message', 'Do you really want to
delete selection?', deleteRecordU);
        }
        else
        {
            Ext.MessageBox.alert('Message', 'Please select at least
one item to delete');
        }//end if/else block
    } // end handleDelete

    function handleEditU(editEvent) {
        var gridField = editEvent.field;
        updateDBU(editEvent);
    }

    function refreshGridU() {
        dsUsers.reload();
    }

    function updateDBU(oGrid_Event) {

        if (oGrid_Event.value instanceof Date) {
            var fieldValue = oGrid_Event.value.format('Y-m-d H:i:s');
        } else
        {
            var fieldValue = oGrid_Event.value;
        }

        //submit to server
        Ext.Ajax.request(
                {
                    waitMsg: 'Saving changes...',
                    url: 'admin.php',
                    params: {
                        task: "updateUser",
                        key: 'UserId',
                        keyID: oGrid_Event.record.data.UserId,
                        field: oGrid_Event.field, //the column name
                        value: fieldValue, //the updated value
                        originalValue: oGrid_Event.record.modified

                    }, //end params

                    failure: function(response, options) {
                        Ext.MessageBox.alert('Warning', 'Oops...');
                    }, //end failure block

                    success: function(response, options) {
                        if (oGrid_Event.record.data.UserId == 0) {
                            var responseData =
Ext.util.JSON.decode(response.responseText);

                            //Extract the ID provided by the server
                            var newID = responseData.newID;
                            //oGrid_Event.record.id = newID;
```

```
                                //Reset the indicator since update
succeeded
                                oGrid_Event.record.set('newRecord',
'no');

                                //Assign the id to the record
                                oGrid_Event.record.set('UserId', newID);
                                dsUsers.commitChanges();

                            } else {
                                dsUsers.commitChanges();
                            }
                        }//end success block
                    }//end request config
            ); //end request
        } //end updateDBU

        //CREATE THE GRID 3.2
        gridUsers = new Ext.grid.EditorGridPanel({
            clicksToEdit: 2,
            colModel: getColumnModelU(),
            disableSelection: true,
            frame: true,
            height: 650,
            id: 'UsersGrid',
            loadMask: true,
            selModel: new Ext.grid.RowSelectionModel({
                singleSelect: false
            }),
            store: dsUsers,
            title: 'Χρήστες',
            trackMouseOver: true,
            width: 1000,
            //Add a bottom bar
            bbar: new Ext.PagingToolbar({
                pageSize: 20,
                store: dsUsers,
                displayInfo: true,
                displayMsg: 'Displaying topics {0} - {1} of {2}',
                emptyMsg: "No data to display",
                items: []
            }), //end bbar
            tbar: [//Add a top bar
                {
                    text: 'Add Record',
                    tooltip: 'Click to Add a row',
                    iconCls: 'add',
                    handler: addRecordU
                }, '-', //add a separator
                {
                    text: 'Delete Selected',
                    tooltip: 'Click to Delete selected row(s)',
                    handler: handleDeleteU,
                    iconCls: 'remove'
                }, '->', // next fields will be aligned to the right
                {
                    text: 'Refresh',
                    tooltip: 'Click to Refresh the table',
                    handler: refreshGridU,
                    iconCls: 'refresh'
                },'-',{
```

```
                text: '0=user/1=admin'

            }
        ]
    });//end grid ,EditorGridPanel
    //3.3 add listeners
    gridUsers.addListener('afteredit', handleEditU);
}//END FUNCTION BULID GRID Users

var buildGridDimoi = function() {

    function addDimo() {
        var d = new DimoiRecordObj({
            IdDimos: 0,
            Dimos: ''
        });

        gridDimoi.stopEditing();//stops any acitve editing fix
        dsDimoi.insert(0, d); //1st arg is index,2nd arg is
Ext.data.Record[] records
        gridDimoi.startEditing(0, 1); //fix
    } // end addDimo

    //function for deleting records
    function deleteDimo(btn) {
        if (btn == 'yes')
        {
            var selectedRows = gridDimoi.selModel.selections.items;
            var selectedKeys = gridDimoi.selModel.selections.keys;
//returns sel. row's id
            var encoded_keys = Ext.encode(selectedKeys);//encode
array into json
            Ext.Ajax.request(
                    {
                        waitMsg: 'Saving changes...',
                        url: 'admin.php',
                        params: {
                            task: "deleteDimo",
                            id: encoded_keys,
                            key: 'IdDimos'
                        },
                        failure: function(response, options) {
                            Ext.MessageBox.alert('Warning',
'Oops...');
                            dsDimoi.rejectChanges();//undo any
changes
                        },
                        success: function(response, options) {
                            dsDimoi.reload();
                        }
                    } //end Ajax request config
            )// end Ajax request initialization
        }//end if click 'yes' on button
    } // end deleteRecord1
    //handler for deleting records
    function handleDeleteDimo() {

        //returns array of selected rows ids only
        var selectedKeys = gridDimoi.selModel.selections.keys;
        if (selectedKeys.length > 0)
        {
```

```
            Ext.MessageBox.confirm('Message', 'Do you really want to
delete selection?', deleteDimo);
        }
        else
        {
            Ext.MessageBox.alert('Message', 'Please select at least
one item to delete');
        }//end if/else block
    } // end handleDelete

    function handleEditDimo(editEvent) {
        var gridField = editEvent.field;
        updateDBDimo(editEvent);
    }

    function refreshGridDimo() {
        dsDimoi.reload();
    }

    function updateDBDimo(oGrid_Event) {

        if (oGrid_Event.value instanceof Date) {
            var fieldValue = oGrid_Event.value.format('Y-m-d H:i:s');
        } else
        {
            var fieldValue = oGrid_Event.value;
        }

        //submit to server
        Ext.Ajax.request(
                {
                    waitMsg: 'Saving changes...',
                    url: 'admin.php',
                    params: {
                        task: "updateDimo",
                        key: 'IdDimos',
                        keyID: oGrid_Event.record.data.IdDimos,
                        field: oGrid_Event.field, //the column name
                        value: fieldValue, //the updated value
                        originalValue: oGrid_Event.record.modified

                    }, //end params

                    failure: function(response, options) {
                        Ext.MessageBox.alert('Warning', 'Oops...');
                        //dsSindiasmoi.rejectChanges();//undo any
changes
                    }, //end failure block

                    success: function(response, options) {
                        //if this is a new record need special
handling
                        if (oGrid_Event.record.data.IdDimos == 0) {
                            var responseData =
Ext.util.JSON.decode(response.responseText);//passed back from server

                            //Extract the ID provided by the server
                            var newID = responseData.newID;
                            //oGrid_Event.record.id = newID;
```

```
                                //Reset the indicator since update
succeeded
                                oGrid_Event.record.set('newRecord',
'no');

                                //Assign the id to the record
                                oGrid_Event.record.set('IdDimos', newID);
                                dsDimoi.commitChanges();

                                //not a new record so just commit changes
                            } else {
                                //commit changes (removes the red
triangle
                                //which indicates a 'dirty' field)
                                dsDimoi.commitChanges();
                            }
                        }//end success block
                    }//end request config
            ); //end request
        } //end updateDB

        //CREATE THE GRID 3.2
        gridDimoi = new Ext.grid.EditorGridPanel({
            clicksToEdit: 2,
            colModel: getColumnModelDimos(),
            verticalScrollerType: 'paginggridscroller',
            invalidateScrollerOnRefresh: false,
            disableSelection: true,
            frame: true,
            // autoExpandColumn:'onoma',
            height: 650,
            width: 1000,
            id: 'DimoiGrid',
            loadMask: true,
            selModel: new Ext.grid.RowSelectionModel({
                singleSelect: false
            }),
            store: dsDimoi,
            title: 'Δήμοι',
            trackMouseOver: true,
            bbar: new Ext.PagingToolbar({
                pageSize: 20,
                store: dsDimoi,
                displayInfo: true,
                displayMsg: 'Displaying topics {0} - {1} of {2}',
                emptyMsg: "No data to display",
                items: []
            }), //end bbar
            tbar: [//Add a top bar
                {
                    text: 'Add Record',
                    tooltip: 'Click to Add a row',
                    iconCls: 'add',
                    handler: addDimo
                }, '-', //add a separator
                {
                    text: 'Delete Selected',
                    tooltip: 'Click to Delete selected row(s)',
                    handler: handleDeleteDimo,
                    iconCls: 'remove'
                }, '->', // next fields will be aligned to the right
```

```
            {
                text: 'Refresh',
                tooltip: 'Click to Refresh the table',
                handler: refreshGridDimo,
                iconCls: 'refresh'
            }
        ]
    });//end grid ,EditorGridPanel
    //3.3 add listeners
    gridDimoi.addListener('afteredit', handleEditDimo);
}//END FUNCTION BULID GRID Dimoi

var buildGridDE = function() {

    function addRecordDE() {
        var de = new DERecordObj({
            IdDE: 0,
            DimotikiEnotita: '',
            IdDimos: ''

        });

        gridDE.stopEditing();//stops any acitve editing fix
        dsDE.insert(0, de); //1st arg is index,2nd arg is
Ext.data.Record[] records
        gridDE.startEditing(0, 1); //fix
    } // end addRecordC

    //function for deleting records
    function deleteRecordDE(btn) {
        if (btn == 'yes')
        {
            var selectedRows = gridDE.selModel.selections.items;
            var selectedKeys = gridDE.selModel.selections.keys;
//returns sel. row's id
            var encoded_keys = Ext.encode(selectedKeys);//encode
array into json
            Ext.Ajax.request(
                    {
                        waitMsg: 'Saving changes...',
                        url: 'admin.php',
                        params: {
                            task: "deleteDE",
                            id: encoded_keys,
                            key: 'IdDE'   //ok
                        },
                        failure: function(response, options) {
                            Ext.MessageBox.alert('Warning',
'Oops...');
                            dsDE.rejectChanges();//undo any changes
                        },
                        success: function(response, options) {
                            dsDE.reload();
                        }
                    } //end Ajax request config
            )// end Ajax request initialization
        }//end if click 'yes' on button
    } // end deleteRecordC
    //handler for deleting records
    function handleDeleteDE() {
```

```
        //returns array of selected rows ids only
        var selectedKeys = gridDE.selModel.selections.keys;
        if (selectedKeys.length > 0)
        {
            Ext.MessageBox.confirm('Message', 'Do you really want to
delete selection?', deleteRecordDE);
        }
        else
        {
            Ext.MessageBox.alert('Message', 'Please select at least
one item to delete');
        }//end if/else block
    } // end handleDeletec

    function handleEditDE(editEvent) {
        var gridField = editEvent.field;
        updateDBDE(editEvent);
    }

    function refreshGridDE() {
        dsDE.reload();
    }

    function updateDBDE(oGrid_Event) {

        if (oGrid_Event.value instanceof Date) {
            var fieldValue = oGrid_Event.value.format('Y-m-d H:i:s');
        } else
        {
            var fieldValue = oGrid_Event.value;
        }

        //submit to server
        Ext.Ajax.request(
                {
                    waitMsg: 'Saving changes...',
                    url: 'admin.php',
                    params: {
                        task: "updateDE",
                        key: 'IdDE', //ok
                        keyID: oGrid_Event.record.data.IdDE,
                        field: oGrid_Event.field, //the column name
                        value: fieldValue, //the updated value
                        originalValue: oGrid_Event.record.modified

                    }, //end params

                    failure: function(response, options) {
                        Ext.MessageBox.alert('Warning', 'Oops...');
                    }, //end failure block

                    success: function(response, options) {
                        //if this is a new record need special
handling
                        if (oGrid_Event.record.data.IdDE == 0) {
                            var responseData =
Ext.util.JSON.decode(response.responseText);//passed back from server

                            //Extract the ID provided by the server
                            var newID = responseData.newID;
                            //oGrid_Event.record.id = newID;
```

```
                                //Reset the indicator since update
succeeded
                                oGrid_Event.record.set('newRecord',
'no');

                                //Assign the id to the record
                                oGrid_Event.record.set('IdDE', newID);
                                dsDE.commitChanges();

                            } else {
                                dsDE.commitChanges();
                            }
                        }//end success block
                    }//end request config
            ); //end request
        } //end updateDBC

        //CREATE THE GRID 3.2
        gridDE = new Ext.grid.EditorGridPanel({
            clicksToEdit: 2,
            store: dsDE,
            title: 'Δημοτική Ενότητα',
            colModel: ColumnModelDE(),
            verticalScrollerType: 'paginggridscroller',
            invalidateScrollerOnRefresh: false,
            disableSelection: true,
            frame: true,
            height: 650,
            id: 'DEGrid',
            plugins: filters,
            loadMask: true,
            selModel: new Ext.grid.RowSelectionModel({
                singleSelect: false
            }),
            trackMouseOver: true,
            width: 1000,
            //Add a bottom bar
            bbar: new Ext.PagingToolbar({
                pageSize: 50,
                store: dsDE,
                displayInfo: true,
                displayMsg: 'Displaying topics {0} - {1} of {2}',
                emptyMsg: "No data to display",
                items: []
            }), //end bbar
            tbar: [//Add a top bar
                {
                    text: 'Add Record',
                    tooltip: 'Click to Add a row',
                    iconCls: 'add',
                    handler: addRecordDE
                }, '-', //add a separator
                {
                    text: 'Delete Selected',
                    tooltip: 'Click to Delete selected row(s)',
                    handler: handleDeleteDE,
                    iconCls: 'remove'
                }, '->', // next fields will be aligned to the right
                {
                    text: 'Refresh',
```

```
                tooltip: 'Click to Refresh the table',
                handler: refreshGridDE,
                iconCls: 'refresh'
            }
        ]
    });//end grid ,EditorGridPanel

    //3.3 add listeners
    gridDE.addListener('afteredit', handleEditDE);
}//END FUNCTION BULID GRID DimotikiEnotita

var buildGridS = function() {

    function addRecordS() {
        var s = new SinikismoiRecordObj({
            IdSin: 0,
            Sinikismoi: '',
            IdDE: '',
            ArithmosPsifoforon: ''

        });

        gridSin.stopEditing();//stops any acitve editing fix
        dsSin.insert(0, s); //1st arg is index,2nd arg is
Ext.data.Record[] records
        gridSin.startEditing(0, 1); //fix
    } // end addRecordC

    //function for deleting records
    function deleteRecordS(btn) {
        if (btn == 'yes')
        {
            var selectedRows = gridSin.selModel.selections.items;
            var selectedKeys = gridSin.selModel.selections.keys;
//returns sel. row's id
            var encoded_keys = Ext.encode(selectedKeys);//encode
array into json
            Ext.Ajax.request(
                    {
                        waitMsg: 'Saving changes...',
                        url: 'admin.php',
                        params: {
                            task: "deleteSinikismo",
                            id: encoded_keys,
                            key: 'IdSin'   //ok
                        },
                        failure: function(response, options) {
                            Ext.MessageBox.alert('Warning',
'Oops...');
                            dsSin.rejectChanges();//undo any changes
                        },
                        success: function(response, options) {
                            dsSin.reload();
                        }
                    } //end Ajax request config
            )// end Ajax request initialization
        }//end if click 'yes' on button
    } // end deleteRecordC
    //handler for deleting records
    function handleDeleteS() {
```

```
        //returns array of selected rows ids only
        var selectedKeys = gridSin.selModel.selections.keys;
        if (selectedKeys.length > 0)
        {
            Ext.MessageBox.confirm('Message', 'Do you really want to
delete selections?', deleteRecordS);
        }
        else
        {
            Ext.MessageBox.alert('Message', 'Please select at least
one item to delete');
        }//end if/else block
    } // end handleDeletes

    function handleEditS(editEvent) {
        var gridField = editEvent.field;
        updateDBS(editEvent);
    }

    function refreshGridS() {
        dsSin.reload();
    }

    function updateDBS(oGrid_Event) {

        if (oGrid_Event.value instanceof Date) {
            var fieldValue = oGrid_Event.value.format('Y-m-d H:i:s');
        } else
        {
            var fieldValue = oGrid_Event.value;
        }

        //submit to server
        Ext.Ajax.request(
                {
                    waitMsg: 'Saving changes...',
                    url: 'admin.php',
                    params: {
                        task: "updateSin",
                        key: 'IdSin', //ok
                        keyID: oGrid_Event.record.data.IdSin,
                        field: oGrid_Event.field, //the column name
                        value: fieldValue, //the updated value
                        originalValue: oGrid_Event.record.modified

                    }, //end params

                    failure: function(response, options) {
                        Ext.MessageBox.alert('Warning', 'Oops...');
                    }, //end failure block

                    success: function(response, options) {
                        //if this is a new record need special
handling
                        if (oGrid_Event.record.data.IdSin == 0) {
                            var responseData =
Ext.util.JSON.decode(response.responseText);//passed back from server

                            //Extract the ID provided by the server
                            var newID = responseData.newID;
                            //oGrid_Event.record.id = newID;
```

```
                                //Reset the indicator since update
succeeded
                                oGrid_Event.record.set('newRecord',
'no');

                                //Assign the id to the record
                                oGrid_Event.record.set('IdSin', newID);
                                dsSin.commitChanges();

                            } else {
                                dsSin.commitChanges();
                            }
                        }//end success block
                    }//end request config
            ); //end request
        } //end updateDBS

        //CREATE THE GRID 3.2
        gridSin = new Ext.grid.EditorGridPanel({
            clicksToEdit: 2,
            store: dsSin,
            title: 'Συνοικισμοί',
            colModel: ColumnModelS(),
            verticalScrollerType: 'paginggridscroller',
            invalidateScrollerOnRefresh: false,
            disableSelection: true,
            frame: true,
            autoHeight:true,
           // height: 1000,
            id: 'SinikismoiGrid',
            plugins: filters,
            loadMask: true,
            selModel: new Ext.grid.RowSelectionModel({
                singleSelect: false
            }),

            trackMouseOver: true,
            width: 1000,
            //Add a bottom bar
            bbar: new Ext.PagingToolbar({
                pageSize: 30,
                store: dsSin,
                displayInfo: true,
                displayMsg: 'Displaying topics {0} - {1} of {2}',
                emptyMsg: "No data to display",
                items: []
            }), //end bbar
            tbar: [//Add a top bar
                {
                    text: 'Add Record',
                    tooltip: 'Click to Add a row',
                    iconCls: 'add',
                    handler: addRecordS
                }, '-', //add a separator
                {
                    text: 'Delete Selected',
                    tooltip: 'Click to Delete selected row(s)',
                    handler: handleDeleteS,
                    iconCls: 'remove'
                }, '->', // next fields will be aligned to the right
```

```
            {
                text: 'Refresh',
                tooltip: 'Click to Refresh the table',
                handler: refreshGridS,
                iconCls: 'refresh'
            }
        ]
    });//end grid ,EditorGridPanel

    //3.3 add listeners
    gridSin.addListener('afteredit', handleEditS);
}


////////////////////////////////////////////////
//////////////////////END OF GRIDS/////////////////
////////////////////////////////////////////////

renderGridP = function() {
    gridPartys.render('grid');//λεμε στο gridPartys να κανει render
στο grid
};//End function render grid

renderGridC = function() {
    gridCandidates.render('grid');
};

renderGridU = function() {
    gridUsers.render('grid');
};

renderGridDimoi = function() {
    gridDimoi.render('grid');
};

renderGridDE = function() {
    gridDE.render('grid');
};

renderGridS = function() {
    gridSin.render('grid');
};


////////////////////////////////////////////////////////////////
//////////////////////////////5.0 load the store////////////////
////////////////////////////////////////////////////////////////


loadParty = function() {
    dsSindiasmoi.load({
        params: {
            start: 0,
            limit: 20
        }
    });
    gridPartys.getSelectionModel().selectFirstRow();

};//end loadParty

loadCandidates = function() {
```

```
    dsCandidates.load({
        params: {
            start: 0,
            limit: 20
        }
    });
    gridCandidates.getSelectionModel().selectFirstRow();

};//end loadCandidate

loadUsers = function() {
    dsUsers.load({
        params: {
            start: 0,
            limit: 20
        }
    });
    gridUsers.getSelectionModel().selectFirstRow();

};//end loadParty

loadDimoi = function() {
    dsDimoi.load({
        params: {
            start: 0,
            limit: 20
        }
    });
    gridDimoi.getSelectionModel().selectFirstRow();

};

loadDE = function() {
    dsDE.load({
        params: {
            start: 0,
            limit: 20
        }
    });
    gridDE.getSelectionModel().selectFirstRow();

};//end loadDE

loadSin = function() {
    dsSin.load({
        params: {
            start: 0,
            limit: 20
        }
    });
    gridSin.getSelectionModel().selectFirstRow();

};


tabs = new Ext.TabPanel({
    activeTab: 0,
    width: 1300,
    height: 800,
    plain: true,
    defaults: {
```

```
        autoScroll: true,
        autoHeight: true
    },
    items: [
    ]
});


function init() {
    setupDataSourceP();
    setupDataSourceC();
    setupDataSourceU();
    setupDataSourceDimoi();
    DataSourceDimotikiEnotita();
    DataSourceSinikismoi();


    getColumnModelP();
    getColumnModelC();
    getColumnModelU();
    getColumnModelDimos();
    ColumnModelDE();
    ColumnModelS();


    buildGridS();
    buildGridDE();
    buildGridU();
    buildGridP();
    buildGridC();
    buildGridDimoi();




    tabs.render('tab-party');
    tabs.render('tab-candidate');
    tabs.render('tab-users');
    tabs.render('tab-dimoi');
    tabs.render('tab-dimotikiEnotita');
    tabs.render('tab-sin');

    tabs.add(gridCandidates).show();
    tabs.add(gridPartys).show();
    tabs.add(gridUsers).show();
    tabs.add(gridDimoi).show();
    tabs.add(gridDE).show();
    tabs.add(gridSin).show();

    loadParty();
    loadCandidates();
    loadUsers();
    loadDimoi();
    loadDE();
    loadSin();


}
```

```
Ext.onReady(init, myNameSpace.myModule, true);
```

Forma_Eisagogis_Psifodeltiou.php

```
<?php
//σύνδεση με βάση δεδομένων και έλεγχος αν έχει γίνει login
include 'common.php';
dbconnect();
checklogin();

$sql = "SELECT * FROM sindiasmoi";
$result = mysql_query($sql);
$count = mysql_num_rows($result);
?>
<html>
    <head>
        <link rel=stylesheet href="default.css">
        <title>Εισαγωγή Νέου Ψηφοδελτίου</title>
        <meta http-equiv="Content-Type" content="text/html;
charset=utf-8">
    </head>
    <body>
<a href ="logout.php"> Logout</a>
<form method="get" action="forma_eisagogis_psifodeltiou2.php">
    <br>
Επέλεξε Συνδιασμό: <br><br>
<select name="koma" >
<?php
//διαβάζουμε τον πίνακα
//βάζουμε στη μεταβλητή $sidmisi της τρέχουσας σειράς το πεδίο
'sidmisi' και στην $id το 'id'
//και δημιουργούμε ένα dropdown που θα εμφανίζει το sidmisi και θα
στέλνει το id
//προσθέτουμε με αρνητικό id -1,-2 τα άκυρα και τα λευκά ώστε να
μπορούμε να τα ξεχωρίζουμε από τα υπόλοιπα
for ($j = 1; $j <= $count; $j++)
{
$row = mysql_fetch_array($result);
$sidmisi = $row['sidmisi'];
$id=$row['id'];
echo '<option value="'.$id.'">'.$sidmisi.'</option>';
}
?>
    <option value="-1">άκυρο...</option>
     <option value="-2">λευκό...</option>

</select>

<br><br>
<input name = "Anazhthsh" type = "submit" value = "Αναζήτηση">
</form>

<?php mysql_close(); ?>
</body>
</html>
```

Forma_Eisagogis_Psifodeltiou2.php

```php
<?php

include 'common.php';
dbconnect();
checklogin();

// ΔΙΑΒΑΖΟΥΜΕ ΤΙΣ ΜΕΤΑΒΛΗΤΕΣ ΠΟΥ ΕΡΧΟΝΤΑΙ ΜΕ GET Η POST
$komaId = (int) @$_REQUEST['koma']; // ID kommatosdropdown

$ypobolh = (int) @$_REQUEST['ypobolh']; // γίνεται 1 με το
'καταχώρηση' των checkbox
$stayroi = @$_REQUEST['ypopsifios']; // Array me τα checkboxes των
υποψηφίων (key=ypopsifiosid,value=1)

if ($komaId < 0) {  // ΑΝ ΕΙΝΑΙ ΛΕΥΚΟ Η ΑΚΥΡΟ
    dbWrite($komaId);   //κλήση στη dbwrite
} else if ($ypobolh==0) {
    showForm2($komaId);  //ΚΑΛΕΣΕ ΤΗ SHOWFORM2 ΜΕ ΤΟ ID ΤΟΥ
ΣΥΝΔΙΑΣΜΟΥ
} else if (is_array($stayroi)) { //αν υπάρχουν σταυροί
    if (count($stayroi)>2) // οι σταυροί είναι πάνω από 2?
        showForm2($komaId,'Δήλωσες πάνω απο 2 σταυρούς');
    else // υποβολή 1η2 σταυρών
        dbWrite($komaId, $stayroi); //καλέσε τη συναρτηση dbwrite
} else { // υποβολή 0 σταυρών
    dbWrite($komaId);
}


    mysql_close();

    /////// FUNCTIONS

    // ΕΜΦΑΝΙΣΗ ΦΟΡΜΑΣ ΜΕ CHECKBOXES
function showForm2($komaId,$msg='') {
    $query = "SELECT * FROM Candidates WHERE PartyId='$komaId'";
    $result = mysql_query($query);
    ?>
    <html>
    <head>
        <link rel=stylesheet href="default.css">
        <title>Εισαγωγή Σταυρών</title>
        <meta http-equiv="Content-Type" content="text/html;
charset=utf-8">
    </head>
    <body>
    <?php
    echo $msg."<br>";
    echo "UserID=".$_SESSION['Userid'].'<br>';


    ?>
    Επέλεξε υποψήφιο: <br />
    *μπορείς να δηλώσεις μέχρι 2 σταυρούς<br/>
    <form method="get">
    <?php
    echo "<input type='hidden' name='koma' value
='$komaId'></input>";
```

```
    echo "<input type='hidden' name='ypobolh' value ='1'></input>";
    while ($row = mysql_fetch_array($result)) {
        $lastname = $row['LastName'];
        $IdCandidate = $row['IdCandidate'];
        echo "<input type=checkbox
name=ypopsifios[$IdCandidate]>$lastname</input><br>";
    }
    echo' <input name = "submit" type = "submit" value =
"Καταχώρηση">';

    ?>
    </form>
    <form action="forma_eisagogis_psifodeltiou.php">
        <input name = "Akyro" type = "submit" value = "Ακυρο">
    </form>
        <?php

    }

    // υποβολή δεδομένων στη βάση
function dbWrite($komaId,$stayroi=NULL) {


        $userId = @$_SESSION['Userid'];

        // Stayroi
        if (!is_null($stayroi)) { // αν εχουμε σταυρούς προς
καταγραφή
           // κάνουμε καταγραφή των σταυρών
            foreach ($stayroi as $IdIpopsifios => $value) {
                $queryStauroi = "update candidates set stauroi =
stauroi+1 where IdCandidate = $IdIpopsifios";
               // echo $queryStauroi . '<br>';
                $result = mysql_query($queryStauroi);
            }


        }

        // καταγραφη ψηφοδελτιου
        $query = "INSERT INTO psifodeltia (idSindiasmo,iduser) VALUES
($komaId,$userId)";
       mysql_query($query);

        header('Location: forma_eisagogis_psifodeltiou.php');

    }



    ?>
```

Ekloges.php

```
<?php

//σύνδεση με βαση δεδομενων
include 'common.php';
dbconnect();

$dimos = @$_GET['dimos'];
$dimotikienotita = @$_GET['dimotikienotita'];
$sinoikismos = @$_GET['sinoikismos'];


?>
<html>
    <head>
        <link rel=stylesheet href="default.css">
        <meta http-equiv="Content-Type" content="text/html;
charset=utf-8">
    </head>
    <title>Αποτελέσματα Εκλογών</title>
    <body>
        <div style="border-top: 0.5cm solid blueviolet"></div>

<?php



$results = getKommata($dimos,$dimotikienotita,$sinoikismos);
//echo '<hr><pre>'.print_r($results, true).'</pre>';
// βάζουμε το sum  των άκυρων σε μεταβλητή
$akyra = @$results[-1]['sum'];
unset($results[-1]); //εξαίρεσε τα απο το array των αποτελεσμάτων

// βάζουμε το sum των λευκών σε μεταβλητή
$leyka = @$results[-2]['sum'];
unset($results[-2]); //εξαίρεσε τα απο το array των αποτελεσμάτων

$kommata = selectTopParties($results);



echo '<table class=noborder>';

echo '<tr>';
echo '<td  colspan=2 class=filtercontainer>';

// Επιλογή Δήμου
echo '<form method=GET>';
echo 'Δήμος:<br>';
$a = array('0'=>'<Όλοι οι Δήμοι>') + getDimos();
dropdown('dimos', $a, $dimos);
echo "<input type='hidden' name='dimotikienotita' value
='0'></input>";
```

```
echo "<input type='hidden' name='sinoikismos' value ='0'></input>";
echo '</form>';

// Επιλογή Δημοτικής Ενότητας
echo '<form method=GET>';
echo 'Δημοτική Ενότητα:<br>';
$a = array('0'=>'<Όλες οι Ενότητες>') + getDimotikiEnotita($dimos);
dropdown('dimotikienotita', $a, $dimotikienotita);
echo "<input type='hidden' name='dimos' value ='$dimos'></input>";
echo "<input type='hidden' name='sinoikismos' value ='0'></input>";
echo '</form>';

// Επιλογή Συνοικισμού
echo '<form method=GET>';
echo 'Συνοικισμός:<br>';
$a = array('0'=>'<Όλοι οι Συνοικισμοί>') +
getSinoikismos($dimotikienotita);
dropdown('sinoikismos', $a, $sinoikismos);
echo "<input type='hidden' name='dimos' value ='$dimos'></input>";
echo "<input type='hidden' name='dimotikienotita' value
='$dimotikienotita'></input>";
echo '</form>';


echo '</td>';
echo '</tr>';

echo '<tr>';
echo '<td class=container>';
//τυπώνουμε array με στατιστικά κομμάτων
printStatistikaKommaton($kommata);

echo '<td class=container>';
printGenikaStoixeia($dimos, $dimotikienotita, $sinoikismos, $akyra,
$leyka);
echo '</td>';

echo '<td  rowspan=3 class=leftborder>';
//ΤΥΠΩΝΟΥΜΕ ΤΗΝ ΕΙΚΟΝΑ ΤΥΠΟΥ 2(top candidates) ΑΠΟ ΤΟ GRAPH.PHP
echo ' ΥΠΟΨΗΦΙΟΙ ΜΕ ΥΨΗΛΟΤΕΡΑ ΠΟΣΟΣΤΑ <BR>';
echo "<img class=graph
src='graph.php?type=2&dimos=$dimos&dimotikienotita=$dimotikienotita&s
inoikismos=$sinoikismos'></img><BR>";
echo '<td>';

echo '</tr>';
echo '<tr>';
echo '<td colspan=2 class=container>';

//ΤΥΠΩΝΟΥΜΕ ΤΗΝ ΕΙΚΟΝΑ ΤΥΠΟΥ 1(top parties) ΑΠΟ ΤΟ GRAPH.PHP
echo 'ΣΥΝΔΙΑΣΜΟΙ ΜΕ ΥΨΗΛΟΤΕΡΑ ΠΟΣΟΣΤΑ<BR>';
echo "<img class=graph
src='graph.php?type=1&dimos=$dimos&dimotikienotita=$dimotikienotita&s
inoikismos=$sinoikismos'></img><BR>";
// Pie
echo '<br>';
echo "<img class=graph
src='graph.php?type=3&dimos=$dimos&dimotikienotita=$dimotikienotita&s
inoikismos=$sinoikismos'></img><BR>";
echo '</td>';
echo '</tr>';
```

```
echo '</table>';

?>
</body>
</html>
```

Graph.php

```
<?php

include 'common.php';
dbconnect();


$type = @$_GET['type'];//τύπος γραφήματος που θα δημιουργηθεί
$dimos = @$_GET['dimos'];
$dimotikienotita = @$_GET['dimotikienotita'];
$sinoikismos = @$_GET['sinoikismos'];

switch ($type) {
    case 1: // Bars κομμάτων
        $im = drawTopParties($dimos,$dimotikienotita,$sinoikismos);
        break;
    case 2: // Bars υποψηφίων
        $im=  drawTopCandidates();
        break;
    case 3: // Pie κομμάτων
        $im =
drawTopPartiesPiechart($dimos,$dimotikienotita,$sinoikismos);
        break;


}

header("Content-type: image/png");
//Output a PNG image to either the browser or a file
imagepng($im);
 //καταστρέφουμε τον δείκτη στην εικονα για να απελευθερώσουμε πόρους
imagedestroy($im);


function drawTopParties($dimos,$dimotikienotita,$sinoikismos) {

    $imagewidth = 400;       // Συνολικό πλάτος εικόνας (pixels)
    $imageheight = 200;      // Συνολικό ύψος εικόνας (pixels)
    $paddingbottom = 20;     // Κενό κάτω από τις bars
    $paddingtop = 20;        // Κενό πάνω από τις bars
    $paddingleft = 50;       // Κενό αριστερά (και δεξιά) από τις bars
    $barwidth = 30;


    $kommataOla = getKommata($dimos,$dimotikienotita,$sinoikismos);
    unset($kommataOla[-1]);
    unset($kommataOla[-2]);
    $kommata = selectTopParties($kommataOla);

    //ΔΗΜΙΟΥΡΓΟΥΜΕ ΤΗΝ ΕΙΚΟΝΑ im- οριζουμε λευκο φοντο
    $im  = imagecreatetruecolor($imagewidth,$imageheight); //Create a
new true color image
    //ImageAntiAlias($im, true);
```

```
    $back = imagecolorallocate($im, 255, 255, 255); // Allocate a
color for an image -white
    $pen = imagecolorallocate($im, 0, 0, 0); // black
    $gray= imagecolorallocate($im, 192, 192, 192);
    imagefill($im, 0, 0, $back); //Flood fill

   // Υπολογίζω σύνολο ψήφων
    $total = 0;
    foreach ($kommata as $komma)
        $total += $komma['sum'];

    // $x,$y είναι το κάτω αριστερά σημείο της τρέχουσας bar μέσα στο
loop
    // Αρχικά, $x,$y είναι το κάτω αριστερά σημείο της πρώτης bar
    $x = $paddingleft;
    $y = $imageheight - $paddingbottom;

    // Βάζουμε στην $maxsum το sum της ψηλότερης bar (θεωρούμε πάντα
ότι είναι η πρώτη)
    $maxsum = 0;

    // $step είναι τα pixels που προχωράμε από bar σε bar
    $step = ($imagewidth - 2*$paddingleft-
$barwidth)/(count($kommata)-1);

    // Βάζουμε στην $maxheight το ύψος της ψηλότερης bar (θεωρούμε
πάντα ότι είναι η πρώτη)
    $maxheight = $imageheight-$paddingtop-$paddingbottom;

    // Για κάθε bar
    foreach ($kommata as $komma) { //$kommata-
>selectTopParties(array)

        // $sum = Το σύνολο ψήφων για την τρέχουσα bar (τρέχον κόμμα)
        $sum = (float) $komma['sum'];

        if ($maxsum == 0) // An eimaste sto 1o bar και δεν έχει
οριστεί ακόμη το maxsum, το ορίζουμε
            $maxsum  = $sum;


        // Υπολογίζω το πάνω αριστερά σημείο (x1,y1) και το κάτω
δεξιά σημείο (x2,y2) της bar
        // Οι τιμές αυτές πρέπει να είναι ακέραιοι για να περάσουν
στην imagefillrectangle()
        $x1 = (int) $x;
        $y1 = (int) ($y - $maxheight * ($sum/$maxsum));
        $x2 = (int) ($x1 + $barwidth);
        $y2 = (int) $y;


        // Fill την μπάρα
        imagefilledrectangle($im, $x1, $y1, $x2, $y2, $gray);

        // τυπώνω το περίγραμμα
        imagerectangle($im, $x1, $y1, $x2, $y2, $pen);

        // τυπώνω τη συντομευση
        imagefttext($im, 10, 90, $x1-2, $y2-2, $pen, './arialbd.ttf',
$komma['sidmisi']);
```

```
        // τυπώνω τους ψήφους σαν ποσοστό
        imagestring($im, 5, $x1, $y1-20, number_format(
(100.0*$komma['sum'])/$total, 0).'%', $pen);

        // βήμα
         $x += $step;

    }

    return $im;
}

function drawTopPartiesPiechart($dimos,$dimotikienotita,$sinoikismos)
{

    $imagewidth = 400;      // Συνολικό πλάτος εικόνας (pixels)
    $imageheight = 200;     // Συνολικό ύψος εικόνας (pixels)
    $piex0 = 120;
    $piey0 = 100;
    $piedx = 150;
    $piedy = 150;
    $legendx0 = 230;
    $legendy0 = 20;
    $legendstep = 30;

    $kommataOla = getKommata($dimos,$dimotikienotita,$sinoikismos);
    unset($kommataOla[-1]);
    unset($kommataOla[-2]);
    $kommata = selectTopParties($kommataOla);

    //ΔΗΜΙΟΥΡΓΟΥΜΕ ΤΗΝ ΕΙΚΟΝΑ im- ορίζουμε λευκο φοντο
    $im  = imagecreatetruecolor($imagewidth,$imageheight); //Create a
new true color image
    //ImageAntiAlias($im, true);
    $back = imagecolorallocate($im, 255, 255, 255); // Allocate a
color for an image -white
    $pen = imagecolorallocate($im, 0, 0, 0); // black

    // Ορίζουμε τα χρώματα των pies
    $colors = array();
    $i = 0;
    foreach ($kommata as $id=>$komma) {
        $colors[$id] = imagecolorallocate($im, 30+20*$i, 30+20*$i,
30+20*$i);
        $i++;
    }

    imagefill($im, 0, 0, $back); //Flood fill

    // Υπολογίζω σύνολο ψήφων
    $total = (float) 0;
    foreach ($kommata as $komma)
        $total += $komma['sum'];



    $start = 0;
    foreach ($kommata as $id => $komma) {
        $end = $start + ($komma['sum']/$total)*360;
        imagefilledarc($im, $piex0, $piey0, $piedx, $piedy,
(int)$start, (int)$end, $colors[$id], IMG_ARC_PIE);
```

```
        imagearc($im, $piex0, $piey0, $piedx, $piedy, (int)$start,
(int)$end, $pen);
        $start = $end;
    }


    // Υπόμνημα
    $y1 = $legendy0;
    foreach ($kommata as $id => $komma) {
        $x1 = $legendx0;
        $x2 = $x1 + 20;
        $y2 = $y1 + 20;
        imagefilledrectangle($im,$x1,$y1,$x2,$y2,$colors[$id]);
        imagerectangle($im,$x1,$y1,$x2,$y2,$pen);
        imagefttext($im, 10, 0, $x2+5, $y2-2, $pen, './arialbd.ttf',
$komma['sidmisi']);
        $y1 += $legendstep;
    }


    return $im;
}


function drawTopCandidates() {

    $imagewidth = 600;  // πλάτος εικόνας (pixels)
    $imageheight = 400; // ύψος εικόνας
    $paddingbottom = 40; //κάτω κενό
    $paddingtop = 40; //πάνω κενό
    $paddingleft = 40; //αριστερό κενό
    $barwidth = 20; //πάχος μπάρας


    $candidates = selecttopcandidates();



    //ΔΗΜΙΟΥΡΓΟΥΜΕ ΤΗΝ ΕΙΚΟΝΑ im- οριζουμε λευκο φοντο
    $im     = imagecreatetruecolor($imagewidth,$imageheight);
//Create a new true color image
    $back = imagecolorallocate($im, 255, 255, 255); // Allocate a
color for an image -white
    imagefill($im, 0, 0, $back); //Flood fill
    $pen = imagecolorallocate($im, 0, 0, 0); // black
    $gray= imagecolorallocate($im, 192, 192, 192);

    // bars
    $x = $paddingleft;
    $maxy = 0;
    foreach ($candidates as $candidate) {

        if ($maxy == 0) // αν ειμαστε στο 1ο bar
            $maxy = $candidate['stauroi'];

        $y = $imageheight - $paddingbottom;

        $x1 = (int) $x;
        $y1 = (int) ($y - ($imageheight-$paddingtop-
$paddingbottom)*($candidate['stauroi']/(float)$maxy));
        $x2 = (int) ($x1 + $barwidth);
```

```
        $y2 = (int) $y;


        // fill την bar
        imagefilledrectangle($im, $x1, $y1, $x2, $y2, $gray);

        // τυπώνουμε το περίγραμμα της μπάρας
        imagerectangle($im, $x1, $y1, $x2, $y2, $pen);

        // τυπώνουμε τη σύντμηση
         imagefttext($im, 10, 90, $x1-2, $y2-2, $pen,
'./arialbd.ttf', $candidate['LastName']);
        // τυπώνουμε τους ψήφους
        imagestring($im, 5, $x1+$barwidth/2-5, $y1-20,
$candidate['stauroi'], $pen);

        // προχωράμε στην επόμενη μπάρα
         $x += ($imagewidth - 2*$paddingleft-
$barwidth)/(count($candidates)-1);
        // $x += 80;


    }

    return $im;
}


?>
```

Common.php

```
<?php


//συνάρτηση που αρχικοποιει τα sessions
function init() {
    session_name('ekloges');
    session_start();
}


//συνάρτηση για συνδεση με τη β.δ
function dbconnect() {
    init();
    $con = mysql_connect("localhost", "root");
    if (!$con)
        die("error to connect" . mysql_error());

    mysql_select_db("ekloges", $con);
    mysql_query('set names utf-8');
}


//συνάρτηση που ελεγχει αν εχει δημιουργηθεί session
function checklogin() {
    if (!isset($_SESSION['Username']))
        header('Location: login.php');
}
```

```php
//συνάρτηση για τα dropdown στο ekloges.php με ορίσματα το επίπεδο και τα περιχόμενα του
function dropdown($name, $array, $selectedValue=0, $autoSubmit=TRUE)
{

    // Ορίζουμε αν θα έχουμε submit αυτόματα με την επιλογή κάποιας τιμής
    if ($autoSubmit)
        $autoSubmit = 'onChange="this.form.submit();"';
    else
        $autoSubmit = '';

    echo "<select name='$name' $autoSubmit >";
    foreach ($array as $key => $value) {
        // Επιλογή συγκεκριμένης τιμής
        if ($key==$selectedValue)
            $selected = ' selected ';
        else
            $selected = '';
        //
        echo "<option value='$key' $selected>$value</option>";
    }
    echo "</select>";
}


//συνάρτηση που δημιουργει array για τους συνδιασμούς
function getKommata($dimos,$dimotikienotita,$sinoikismos) {

    // Δημιουργώ το φιλτρο
    if ($sinoikismos!=0)
        $where = "WHERE sinoikismoi.IdSin = $sinoikismos";
    else if ($dimotikienotita!=0)
        $where = "WHERE dimotikienotita.IdDE = $dimotikienotita";
    else if ($dimos!=0)
        $where = "WHERE dimos.IdDimos = $dimos";
    else
        $where = '';

    //δημιουργώ array με το id του καθε κόμματος-το sum των ψήφων-τη σύντμιση ταξινομεμένα ανά sum ψήφων φθήνουσα πορεία
    $query = 'select idSindiasmo, count(*) as sum, sindiasmoi.sidmisi as sidmisi
    from psifodeltia
    left join sindiasmoi on sindiasmoi.id =psifodeltia.idsindiasmo
    left join users on users.UserId = psifodeltia.iduser
    left join sinoikismoi on sinoikismoi.IdSin = users.IdS
    left join dimotikienotita on dimotikienotita.IdDE = sinoikismoi.IdDE
    left join dimos on dimos.IdDimos = dimotikienotita.IdDimos
    '.$where.'
    group by idSindiasmo
    order by sum desc';
    $resource = mysql_query($query);

    $result = array();
    while ($row = mysql_fetch_assoc($resource)) { //οσο υπάρχουν αποτελέσματα
            $id = $row['idSindiasmo'];          //βάλε το 'idSindiasmo' της τρεχουσας σειρας στη μεταβλητή $id
```

```
            unset($row['idSindiasmo']);                 //μη το
εμφανίσεις στο array
            $result[$id]=$row;                          //βάλε το id της
τρέχουσας σειράς στο array $result
    }
    return $result;
}


//συνάρτηση για να ξεχωρίσουμε τα κόμματα με τους περισσότερους
ψήφους
function selectTopParties($kommata) {
    // οι συνδιασμοί είναι ταξινομιμένοι με φθίνουσα σειρά ψήφων
    $top = 5; //πόσα κόμματα θέλουμε
    $result = array();// δημιούργησε κενό array
    $i=0;
    $sumloipa = 0; //μεταβλητή για να προσθετουμε τις ψήφους των
λοιπών πέρα απο τα top
    foreach ($kommata as $id => $komma) {
        $i++;
        if ($i<=$top) { //αν δεν έχεις ξεπεράσει τα 5 κορυφαία
κόμματα
            $result[$id]=$komma; //βάλε το id του κόμματος στο
$result
        } else {
            $sumloipa += $komma['sum']; //αλλιώς αύξησε τη μεταβλητή
sumloipa κατα τους ψηφους που εχει το τρέχον κόμμα
        }
    }

    // Έχουμε λοιπά?
    if ($sumloipa>0) { // αν ναι
    //με τη τιμή -3 ορίζουμε το συνολο των ψηφων των κομμάτων και στο
πεδιο sidmisi εμφάνισε λοιπά
        $result[-3] = array('sum'=>$sumloipa, 'sidmisi'=>'ΛΟΙΠΑ' );

    }

    return $result;
}


function selecttopcandidates()
{
$query='select IdCandidate,LastName,stauroi
        from candidates
        order by stauroi desc';
$resource = mysql_query($query);
$result = array();
for ($i = 1; $i <= 10; $i++)
{
$row = mysql_fetch_assoc($resource);

            $id = $row['IdCandidate']; // βάλε της τρέχουσας σειράς
το idcandidate στη μεταβλητή id
             unset($row['IdCandidate']);
            $result[$id]=$row; //βάλε στο array $result το id της
τρέχουσας σειράς
}
    return $result;
```

```
}


function getDimos() {

    $query = 'select *
    from dimos
    order by Dimos';
    $resource = mysql_query($query);

    $result = array();
    while ($row = mysql_fetch_assoc($resource)) {
            $id = $row['IdDimos'];
            $onoma = $row['Dimos'];
            $result[$id] = $onoma;
    }
    return $result;
}

function getDimotikiEnotita($dimos=0) {

    if ($dimos!=0)
        $where = "WHERE dimos.IdDimos = $dimos";
    else
        $where = '';

    $query = 'select dimotikienotita.*
    from dimotikienotita
    left join dimos on dimos.IdDimos = dimotikienotita.IdDimos '
    .$where
    .' order by DimotikiEnotita';
    $resource = mysql_query($query);

    $result = array();
    while ($row = mysql_fetch_assoc($resource)) {
            $id = $row['IdDE'];
            $onoma= $row['DimotikiEnotita'];
            $result[$id] = $onoma;
    }
    return $result;
}

function getSinoikismos($dimotikienotita=0) {

    if ($dimotikienotita!=0)
        $where = "WHERE dimotikienotita.IdDE = $dimotikienotita";
    else
        $where = '';

    $query = 'select *
    from sinoikismoi
    left join dimotikienotita on dimotikienotita.IdDE =
sinoikismoi.IdDE '
    .$where
    .' order by Sinikismoi';
    $resource = mysql_query($query);

    $result = array();
    while ($row = mysql_fetch_assoc($resource)) {
            $id = $row['IdSin'];
            $onoma= $row['Sinikismoi'];
```

```
            $result[$id] = $onoma;
    }
    return $result;
}

function printGenikaStoixeia($dimos, $dimotikienotita, $sinoikismos,
$akyra, $leyka) {

    // Δημιουργώ το φιλτρο
    if ($sinoikismos!=0)
        $where = "WHERE sinoikismoi.IdSin = $sinoikismos";
    else if ($dimotikienotita!=0)
        $where = "WHERE dimotikienotita.IdDE = $dimotikienotita";
    else if ($dimos!=0)
        $where = "WHERE dimos.IdDimos = $dimos";
    else
        $where = '';


    // Υπολογίζουμε αριθμό ψηφοφόρων
    $query = 'SELECT sum( ArithmosPsifoforon ) as psifoforoi FROM
sinoikismoi
        left join dimotikienotita on dimotikienotita.IdDE =
sinoikismoi.IdDE
        left join dimos on dimos.IdDimos = dimotikienotita.IdDimos '
    .$where;
    $resource = mysql_query($query);
    $row = mysql_fetch_assoc($resource);
    $ArithmosPsifoforon = $row['psifoforoi'];

    // Υπολογίζουμε Αριθμό Ψηφισάντων και Αποχή

    $query = 'SELECT count(*) as countpsifodeltia FROM psifodeltia
        left join users on users.UserId = psifodeltia.iduser
        left join sinoikismoi on sinoikismoi.IdSin = users.IdS
        left join dimotikienotita on dimotikienotita.IdDE =
sinoikismoi.IdDE
        left join dimos on dimos.IdDimos = dimotikienotita.IdDimos '
        .$where;

    $resource = mysql_query($query);
    $row = mysql_fetch_assoc($resource);
    $ArithmosPsifisanton = $row['countpsifodeltia'];
    $apoxi = $ArithmosPsifoforon - $ArithmosPsifisanton;

    // Τυπώνουμε array με γενικά στοιχεία
    echo "<table class=report>";
    echo "<tr><td class='title'>ΕΓΓΕΓΡΑΜΜΕΝΟΙ:</td><td
class='right'>$ArithmosPsifoforon</td></tr>";
    echo "<tr><td class='title'>ΑΠΟΧΗ:</td><td
class='right'>$apoxi</td></tr>";
    echo "<tr><td class='title'>ΑΚΥΡΑ:</td><td
class='right'>$akyra</td></tr>";
    echo "<tr><td class='title'>ΛΕΥΚΑ:</td><td
class='right'>$leyka</td></tr>";
    echo "</table>";

}

function printStatistikaKommaton($kommata) {
    echo "<table class=report>";
```

```
    echo "<tr class=title><td class='center'>Κόμμα</td><td
class='right'>Ψηφοδέλτια</td></tr>";
    foreach ($kommata as $komma) {
        $sidmisi = $komma['sidmisi'];
        $sum = $komma['sum'];
        echo "<tr><td class='center'>$sidmisi</td><td
class='right'>$sum</td></tr>";
    }
    echo "</table>";
}


?>
```

Login.php

```
<html>
    <head>
        <link rel=stylesheet href="default.css">
        <title>Login</title>
        <meta http-equiv="Content-Type" content="text/html;
charset=utf-8">
    </head>
    <body>
<h1> Login</h1>

<form class='loginform' method="POST" action="checklogin.php">
<table border="0">
<tr><td>Username</td><td>:</td><td><input type="text" name="Username"
size="20"></td></tr>
<tr><td>Password</td><td>:</td><td><input type="password"
name="Password" size="20"></td></tr>
<tr><td> </td><td> </td><td><input type="submit"
value="Login"></td></tr>
</table>
</form>

</body>

</html>
```

Checklogin.php

```
<?php

//Inialize session
include 'common.php';
dbconnect();

// Retrieve username and password from database according to user's
input
$login = mysql_query("SELECT * FROM users WHERE (Username = '" .
mysql_real_escape_string($_POST['Username']) . "') and (Password = '"
. mysql_real_escape_string($_POST['Password']) . "')");
// Check username and password match
// Mysql_num_row is counting table row
$count=mysql_num_rows($login);
if ($count == 1) { //an vrethei
```

```
// Set username session variable
$_SESSION['Username'] = $_POST['Username'];
$user = mysql_fetch_assoc($login);
$_SESSION['Userid'] = $user['UserId'];
session_write_close();
if ($user['OmadaErgasias']) {     //check if user is admin
  // Jump to secured page
//echo '<pre>'.print_r($user, true).'</pre>';
//die();

    header('Location: administrator.php');
}
else
  header('Location: forma_eisagogis_psifodeltiou.php');
}
else  {
// Jump to login page
    ?>
    <html>
    <head>
        <link rel=stylesheet href="default.css">
        <title>Λαθός Κωδικός ή Όνομα Χρήστη</title>
        <meta http-equiv="Content-Type" content="text/html;
charset=utf-8">
    </head>
    <body>
     ΤΟ USERNAME Η Ο ΚΩΔΙΚΟΣ ΠΟΥ ΔΩΣΑΤΕ ΕΙΝΑΙ ΛΑΘΟΣ
     <br><a href ='login.php'>ΕΠΙΣΤΡΟΦΗ ΣΤΗ ΠΡΟΗΓΟΥΜΕΝΗ ΦΟΡΜΑ</a>
    </body>
    </html>

<?php
}

?>
```

Logout.php

```
<?php

// Inialize session
include 'common.php';
init();

// Delete certain session
unset($_SESSION['Username']);
// Delete all session variables
// session_destroy();
session_write_close();

// Jump to login page
header('Location: login.php');

?>
```

Default.css

```
td {
    border: 1px solid #c0c0c0;
}

table {
    border: 2px solid black;
    border-collapse: collapse;
}

.loginform table {
    border: 1px solid rgba(0,0,0,0.2);
    margin: 0px auto;
    background-color: rgba(255,255,255,0.5);


}

.loginform td {
    border: 0;
    padding: 10px;

}

.report td {
    padding: 10px;
}

.left {
    text-align: left;
}
.center {
    text-align: center;
}
.right {
    text-align: right;
}

.title td {
    font-weight: bold;
    border-bottom: 1px solid black;
}

img.graph {
    margin: 30px;
    padding:0px;
    border:1px solid rgba(0,0,0,0.5);
    border-radius: 15px;
    box-shadow: 1px 1px 4px rgba(0,0,0,0.3);

}

body {background-image:url('shl.png');}

.noborder { border: 0;}

.leftborder {
    border: 0;
    border-left: 1px solid rgba(0,0,0,0.5);
    padding: 20px;
    text-align: center;
```

```
    vertical-align: top;
}

.container {
    border: 0;
    padding: 20px;
    text-align: center;
    vertical-align: top;
}

.filtercontainer {
    border: 0;
}

.filtercontainer form {
    display: block;
    float: left;
    padding-left: 20px;
}
```